UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

USO DE TÉCNICAS DE APRENDIZAGEM PROFUNDA NA CLASSIFICAÇÃO DE CONFIGURAÇÕES DE MÃO DE LÍNGUA DE SINAIS

ANNE DE SOUZA OLIVEIRA

MANAUS
2019

UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ANNE DE SOUZA OLIVEIRA

USO DE TÉCNICAS DE APRENDIZAGEM PROFUNDA NA CLASSIFICAÇÃO DE CONFIGURAÇÕES DE MÃO DE LÍNGUA DE SINAIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, área de concentração Controle e Automação de Sistemas.

Orientadora: Profa. Dra. Marly Guimarães Fernandes Costa
Coorientador: Prof. Dr. Cícero Ferreira Fernandes Costa Filho

MANAUS
2019

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Oliveira, Anne de Souza
O48u Uso de Técnicas de Aprendizagem Profunda na Classificação de Configurações de Mão de Língua de Sinais / Anne de Souza Oliveira. 2019
96 f.: il. color; 31 cm.

Orientadora: Marly Guimarães Fernandes Costa
Coorientador: Cícero Ferreira Fernandes Costa Filho
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Amazonas.

1. língua brasileira de sinais. 2. redes neurais convolucionais. 3. reconhecimento de sinais. 4. aprendizagem profunda. 5. técnicas de regularização. I. Costa, Marly Guimarães Fernandes II. Universidade Federal do Amazonas III. Título

ANNE DE SOUZA OLIVEIRA

USO DE TÉCNICAS DE APREDIZAGEM PROFUNDA NA CLASSIFICAÇÃO DE CONFIGURAÇÕES DE MÃO DE LÍNGUA DE SINAIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica na área de concentração Controle e Automação de Sistemas.

Aprovado em 03 de janeiro de 2019.

BANCA EXAMINADORA

Profª. Dra. Marly Guimarães Fernandes Costa, Presidente

Universidade Federal do Amazonas

Prof. Dr. José Raimundo Gomes Pereira, Membro

Universidade Federal do Amazonas

Prof. Dr. Jozias Parente de Oliveira, Membro

Universidade do Estado do Amazonas

A minha mãe, mulher amável, que sempre me inspira e apoia.

AGRADECIMENTOS

Aos deuses, por terem me dado saúde, determinação e principalmente equilíbrio para superar todos os obstáculos encontrados.

À minha família por ser meu suporte de incentivo e amparo em todas as esferas da vida. Sem vocês, sem o seu amor, teria sido muito mais difícil.

À minha orientadora e meu coorientador por partilharem seu conhecimento, por disponibilizarem um local onde fosse possível imergir na pesquisa e por contribuírem significativamente todas as vezes que foi preciso.

Aos meus amigos que partilharam dos anseios, das dúvidas, das certezas e de todas as oscilações que essa experiência foi capaz de proporcionar a cada um de nós. Partilhar esses momentos com vocês, sem dúvida, tornou a trajetória mais suave.

Ao Centro de Pesquisa e Desenvolvimento de Tecnologia Eletrônica e da Informação - CETELI da Universidade Federal do Amazonas - UFAM por propiciar a infraestrutura necessária à realização desta dissertação.

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

"Que o homem nunca esqueça que a verdadeira inteligência humana está na capacidade de se colocar no lugar do outro".

Anne Oliveira (2018)

RESUMO

Este trabalho apresenta a utilização de redes neurais convolucionais na classificação das configurações de mão da língua brasileira de sinais. Para elaborar modelos com capacidade de aprendizagem relacionada a essa língua, foi utilizado o conjunto de dados *LibrasImage*. As arquiteturas de redes utilizadas foram selecionadas com base na pesquisa bibliográfica sistemática realizada. Diferentes valores de hiperparâmetros foram testados para verificação e escolha daqueles que melhor se adequassem a tarefa de classificação. Os treinamentos dos modelos foram realizados por 500 épocas com três arquiteturas diferentes e duas técnicas de regularização (*dropout* e L2). Para testar o desempenho dos doze modelos com relação a classificação das configurações de mão, a acurácia foi a medida de desempenho escolhida para comparação. Para cada uma das arquiteturas, o modelo com maior acurácia foi selecionado para ser analisado com relação a sensibilidade, área sob a curva ROC e taxa de erro para cada uma das configurações de mão presentes no conjunto de dados *LibrasImage*. O modelo com melhor desempenho com relação as medidas citadas, foi comparado ao modelo treinado com o classificador k-vizinhos mais próximos, apresentado no trabalho de Costa Filho et al. (2017), para diferentes medidas de avaliação: acurácia, sensibilidade, precisão e F1 *score*. Os resultados mostraram que a utilização de redes neurais convolucionais é uma técnica que melhora a aprendizagem das configurações de mão da língua brasileira de sinais em relação ao outro método de classificação disponível na literatura que foi testado com o mesmo conjunto de dados, apresentando uma acurácia de 97,98%. A diferença de desempenho entre os dois métodos, em termos de acurácia, foi avaliada com o teste qui-quadrado de Pearson, cujo resultado mostrou ser estatisticamente significativo.

Palavras-chave: língua brasileira de sinais, redes neurais convolucionais, reconhecimento de sinais, aprendizagem profunda, técnicas de regularização, *dropout*, L2, classificação.

ABSTRACT

This work presents a method to classify Brazilian sign language hand configurations using convolutional neural networks. The network architectures used were selected based on a systematic bibliographic research. Several experiments were done using different values of hyperparameters aiming to obtain the best fit the classification task. The models training was carried out for 500 epochs using three different architectures and two regularization techniques (dropout and L2). LibrasImage, a data set of hand configurations depth images was used in the training and testing steps of the models. The models were analyzed with respect to the accuracy, sensitivity, area under the ROC curve and error rate for each hand configuration. The best result obtained was an accuracy of 97.98%. This result shows that the use of convolutional neural network improves the classification of Brazilian sign language hand configurations in relation to the method that uses the k-nearest neighbor classifier, that was tested with the same dataset. The difference in performance between the two methods was statistically significant by Pearson chi-square test.

Keywords: Brazilian sign language, convolutional neural networks, sign recognition, deep learning, regularization techniques, dropout, L2, classification.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

$(2D)^2PCA$	*Two-Directional Two-Dimensional Principal Component Analysis*
ASL	*American Sign Language*
AUC	*Area Under Curve*
CBEB	Congresso Brasileiro de Engenharia Biomédica
CETELI	Centro de Tecnologia Eletrônica e da Informação
CNN	*Convolutional Neural Networks*
CPU	*Central Processing Unit*
DNN	*Deep Neural Network*
GPU	*Graphics Processing Unit*
HMM	*Hidden Markov Model*
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
kNN	*k-Nearest Neighbors*
LCN	*Local Contrast Normalization*
Libras	Língua Brasileira de Sinais
LOOT	*Leave One Out Training and Testing*
MDP	*Maximum Diversity Problem*
PR&I Lab	Laboratório de Reconhecimento de Padrões e Otimização
RAM	*Random Access Memory*
RBF	*Radial Basis Function*
ReLU	*Rectified Linear Unit*
RGB	*Red Green Blue*
SVM	*Support Vector Machine*
UFAM	Universidade Federal do Amazonas

SUMÁRIO

1 INTRODUÇÃO

Para se comunicar, a comunidade surda utiliza a língua de sinais. Similar ao que ocorre com a língua falada, não há uma língua de sinais universal. Entre os sinais existentes em uma língua de sinais estão os estáticos e os que envolvem movimento (ALMEIDA; GUIMARAES; RAMIREZ, 2014; PORFIRIO et al., 2013).

A classificação desses sinais por meio de técnicas de aprendizagem de máquina não é uma tarefa trivial, pois alguns sinais podem ser parecidos uns com os outros, com apenas a posição do dedo polegar os diferenciando. Outra variação acontece pela maneira com que cada indivíduo realiza um mesmo sinal (HUANG et al., 2015; COSTA FILHO et al., 2017; COSTA FILHO et al., 2016).

O aprendizado de máquina é uma ciência que dá aos computadores a habilidade de aprender a partir de exemplos. Entre as tarefas de aprendizado de máquina existentes, esse trabalho foca na tarefa de classificação, que consiste em determinar a categoria que determinada entrada pertence (GÉRON, 2017).

Para que um classificador desempenhe sua tarefa, o mesmo deve passar por um processo de aprendizagem. Entre as categorias de aprendizado existentes, esse trabalho utiliza aprendizado supervisionado (GOODFELLOW; BENGIO; COURVILLE, 2016), que é aquele realizado através de métodos de otimização, comparando a saída da máquina com um resultado esperado.

Para que o processo de aprendizagem possa ocorrer é necessário fornecer um conjunto de treinamento, com exemplos pertencentes a cada classe (DANGETI, 2017). Dentre as técnicas de aprendizagem de máquina, este trabalho utiliza a aprendizagem profunda com arquiteturas de redes neurais convolucionais.

Uma rede neural profunda é uma rede neural com duas ou mais camadas escondidas (PATTERSON; GIBSON, 2017). Diferentemente das redes neurais tradicionais, as redes neurais convolucionais possuem diferentes tipos de camadas e têm, como uma das suas principais características, o fato de realizarem, automaticamente, a extração de características.

As redes neurais convolucionais são um tipo de arquitetura de redes neurais profundas especializadas no processamento de dados em formato de grade (BUDUMA; LOCASCIO, 2017), como as imagens. Para gerar as ativações de uma camada, as mesmas aplicam a operação de convolução, seguida por uma função de ativação e agrupamento. Essas operações podem ser realizadas várias vezes para extração das características que serão classificadas posteriormente.

Normalmente, é difícil dimensionar a capacidade apropriada de um modelo para resolver um determinado problema. Assim, para evitar problemas de *underfitting* (KIM, 2017), impossibilidade de representar as relações entre os dados de entrada e saída no conjunto de treinamento, trabalha-se, normalmente, com modelos superdimensionados. Tais modelos, embora aproximem bem no conjunto de treinamento, podem gerar problemas de *overfitting* (HOPE; RESHEFF; LIEDER, 2017), impossibilidade de representar as relações entre os dados de entrada e saída no

conjunto de teste.

Para se evitar problemas de *overfitting*, empregam-se normalmente técnicas de regularização. A maioria das técnicas de regularização é baseada na regularização de estimadores, que consiste em reduzir a variância sem aumentar a polarização (*bias*) (SUGOMORI et al., 2017).

Uma das formas de fazer um modelo generalizar melhor é penalizando pesos grandes com altos valores. Uma forma de realizar essa penalização é através da técnica L2 (GOODFELLOW; BENGIO; COURVILLE, 2016), também conhecida como decaimento de pesos. Essa técnica consiste em introduzir um termo dependente do valor dos pesos na função objetivo.

Outra maneira é treinando simultaneamente vários modelos de redes neurais, através da omissão aleatória de unidades da camada escondida. Na hora de testar um exemplo, considera-se a média aritmética ou geométrica da saída desses vários modelos. Essa técnica é conhecida como *dropout* (GOODFELLOW; BENGIO; COURVILLE, 2016).

Com relação à classificação da língua de sinais, baseada na revisão realizada, alguns trabalhos têm explorado a utilização de redes neurais convolucionais para classificar a língua americana de sinais como os de Li et al. (2015), Zhu et al. (2016), Kang, Tripathi e Nguyen (2015) e Ameen e Vadera (2017) e um trabalho tem explorado a utilização de redes neurais convolucionais na classificação da língua italiana, o de Pigou et al. (2015)

Há ainda um trabalho que aplicou redes neurais convolucionais para classificar símbolos do sistema de escrita *SignWriting* relacionados aos sinais pertencentes a língua brasileira de sinais, o de Stiehl et al. (2015). O *SignWriting* inclui os símbolos usados para escrever as formas da mão, movimentos, expressões faciais e gestos corporais que foram representados neste trabalho por esqueletos da imagem.

Diferentemente do trabalho de Stiehl et al. (2015), neste trabalho, foi verificado se a utilização de redes neurais convolucionais junto com as técnicas de regularização melhora os resultados obtidos na classificação das configurações de mão da língua brasileira de sinais representadas em imagens de profundidade obtidas por um sensor *Kinect*.

1.1 Objetivos

O objetivo geral deste trabalho é avaliar se, a utilização de redes neurais convolucionais, é uma técnica que melhora o desempenho em relação a técnica tradicional, apresentada no trabalho de Costa Filho et al. (2017) com o classificador k-vizinhos mais próximos, com relação a classificação das configurações de mão da língua brasileira de sinais.

Os objetivos específicos deste trabalho são:

- Avaliar o desempenho dos modelos treinados por três arquiteturas de redes neurais convolucionais com diferentes níveis de profundidade;
- Avaliar se a utilização da técnica de regularização *dropout* aumenta a capacidade de generalização dos modelos treinados com redes neurais convolucionais;

- Avaliar se a utilização da técnica de regularização L2 aumenta o desempenho dos modelos treinado com redes neurais convolucionais;
- Avaliar se a combinação das técnicas de regularização *dropout* + L2 aumenta a capacidade de classificação dos modelos gerados com as arquiteturas de redes neurais convolucionais;
- Analisar os três melhores desempenhos dos modelos gerados com as arquiteturas de redes neurais convolucionais em relação à classificação das configurações de mão da língua brasileira de sinais;
- Comparar o modelo com melhor desempenho treinado com rede neural convolucional com o modelo treinado pelo classificador k-vizinhos mais próximos presente no trabalho de Costa Filho et al. (2017).

1.2 Organização do trabalho

Este trabalho possui cinco capítulos e três apêndices relacionados com a utilização de redes neurais convolucionais para obtenção de modelos com capacidade de classificar as configurações de mão pertencentes a língua brasileira de sinais.

O capítulo 2 de REVISÃO BIBLIOGRÁFICA é resultado de uma pesquisa bibliográfica sistemática realizada nas bases de dados: Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), *Web of Science* e *Engineering Village* disponível no Apêndice PESQUISA BIBLIOGRÁFICA SISTEMÁTICA. Três questões de pesquisa foram inicialmente elaboradas e cada questão foi transformada em uma composição de palavras chave e estratégia de busca. Cada composição foi adaptada a forma de busca utilizada por cada uma das bases de dados.

O total de artigos recuperados foram 38, excluindo as repetições, dos quais apenas 14 foram mantidos após a leitura de triagem. Na fase de análise observou-se que todos os artigos respondiam a um dos questionamentos levantados no início da pesquisa e, por esse motivo, todos foram mantidos e são apresentados neste capítulo de revisão.

O capítulo 3 de FUNDAMENTAÇÃO TEÓRICA é resultado das pesquisas realizadas em meios disponibilizados digitalmente sobre aprendizagem profunda, focada nas redes neurais convolucionais. Alguns conceitos precisavam ficar bem elucidados, como a diferença entre as redes neurais convolucionais e as tradicionais, assim como o entendimento sobre o funcionamento de cada uma das camadas constituintes das redes neurais convolucionais profundas. Além disso, conceitos relacionados a utilização de algoritmo de aprendizagem, ajuste de hiperparâmetros e utilização de técnicas de regularização também são abordados neste capítulo de fundamentação.

O capítulo 4 de MATERIAIS E MÉTODOS é resultado de vários procedimentos que precisavam ser planejados para que os modelos de classificação das configurações de mão da língua brasileira de sinais fossem obtidos. Para realização dos treinamentos, foi definida a utilização de dois computadores e realizado um levantamento sobre a configuração dos mesmos.

Dentre os trabalhos revisados, foi escolhido um que disponibilizasse o conjunto de dados. Dessa forma, seria possível avaliar o desempenho do classificador adotado em relação às redes neurais convolucionais. O trabalho de Costa Filho et al. (2017) foi o que apresentou essa característica. A partir da escolha do trabalho, optou-se por usar o mesmo conjunto de dados pré-processado e a mesma divisão do conjunto de dados.

As arquiteturas escolhidas foram definidas com base nos trabalhos que usaram redes convolucionais, que foram apresentados na revisão bibliográfica. O otimizador e o tipo de armazenamento foram baseados no ambiente de aprendizagem adotado. A etapa de ajuste de hiperparâmetros foi definida para que fossem encontrados os valores mais adequados para cada arquitetura.

Os treinamentos foram planejados com o objetivo de caracterizar o desempenho de cada arquitetura e de cada técnica de regularização. A escolha da medida de avaliação usada no teste dos modelos foi baseada na métrica comum empregada em todos os artigos revisados. Para comparação com o modelo apresentado no trabalho de Costa Filho et al. (2017), foram definidas etapas e medidas de avaliação a serem empregadas para seleção do modelo com melhor desempenho.

O capítulo 5 de RESULTADOS E DISCUSSÕES apresenta o desempenho dos modelos obtidos a partir do treinamento das arquiteturas selecionadas e o desempenho dos modelos obtidos por essas arquiteturas utilizando técnicas de regularização, totalizando doze modelos.

Todos os doze modelos foram avaliados conforme sua capacidade de generalização relacionada a configuração de mão da língua brasileira de sinais. Para cada arquitetura, foi selecionado um modelo com melhor capacidade de generalização, totalizando três modelos, e esses modelos foram analisados conforme as classificações relacionadas as sessenta e uma configurações de mão da língua brasileira de sinais.

O modelo com o melhor desempenho foi, então, comparado ao resultado apresentados no trabalho de Costa Filho et al. (2017) para o classsificador k-vizinhos mais próximos. Um teste de significância estatística foi aplicado para comparar a diferença de desempenho dos referidos modelos.

O Apêndice A - PESQUISA BIBLIOGRÁFICA SISTEMÁTICA, apresenta a pesquisa bibliográfica realizada após definição do tema, para compreensão do estado da arte relacionado à classificação da língua de sinais. O Apêndice B - MATRIZ DE CONFUSÃO apresenta as matrizes de confusão relacionadas aos maiores desempenhos dos modelos treinados com redes neurais convolucionais. O Apêndice C - ARTIGO disponibiliza o artigo produzido decorrente deste trabalho:

- *"Recognizing Hand Configurations of Brazilian Sign Language using Convolutional Neural Networks"* apresentado no XXVI Congresso Brasileiro de Engenharia Biomédica (CBEB 2018), realizado na cidade de Búzios, Rio de Janeiro, de 21 a 25 de outubro de 2018.

2 REVISÃO BIBLIOGRÁFICA

Para verificar se a utilização de redes neurais convolucionais já era uma técnica adotada na classificação das configurações de mão da língua brasileira de sinais, foi necessário realizar uma pesquisa bibliográfica sistemática a respeito do tema. Neste capítulo, são descritos os passos executados para realização da pesquisa e os resultados advindos da mesma.

2.1 Pesquisa bibliográfica

O apêndice A - PESQUISA BIBLIOGRÁFICA SISTEMÁTICA apresenta as fases de planejamento e condução da pesquisa bibliográfica realizada para este trabalho. Na fase de planejamento, houve a definição do tema, elaboração de três questões de pesquisa e definição das bases de dados onde seriam realizadas as consultas: IEEE, *Web of Science* e *Engineering Village*. Cada questão elaborada foi transformada em uma composição de palavras chave e estratégia de busca, adaptadas a forma de busca utilizada por cada uma das bases de dados.

Na fase de condução, a execução da pesquisa nas bases de dados retornou 38 artigos, excluindo as repetições. Os artigos passaram por duas etapas: triagem e análise. Na etapa de triagem foram executadas as seguintes atividades:

- Definição dos critérios de inclusão de um artigo para revisão baseados nas questões elaboradas;
- Leitura generalizada do artigo para identificar a relação deste com as questões;
- Decisão de manter ou eliminar o artigo de acordo com os critérios estabelecidos.

Na etapa de análise foi realizada uma leitura mais detalhada de cada artigo selecionado na triagem. Nessa etapa, como todos os artigos respondiam a um dos questionamentos levantados no início da pesquisa, todos foram mantidos e estão apresentados na seção 2.2 deste capítulo.

2.2 Aprendizagem de máquina no reconhecimento de língua de sinais

O reconhecimento de língua de sinais vem sendo praticado através de dois ramos da aprendizagem de máquina identificados nesta revisão:

- Aprendizagem tradicional;
- Aprendizagem profunda.

A aprendizagem tradicional envolve uma etapa de pré-processamento de imagem para extração de características e tem sido utilizada para classificação da língua brasileira de sinais. A seção 2.2.1 apresenta os trabalhos que relacionam a aprendizagem tradicional e a língua brasileira de sinais.

A aprendizagem profunda consegue trabalhar diretamente com imagens, sem a necessidade de uma etapa de pré-processamento, e foi utilizada em um trabalho de classificação da língua brasileira de sinais apresentado na seção 2.2.2. Outras línguas de sinais, como a americana e a italiana, também tem utilizado a aprendizagem profunda na classificação de imagens. A seção 2.2.3 apresenta os trabalhos que relacionam a aprendizagem profunda e as línguas de sinais americana e italiana.

2.2.1 Aprendizagem tradicional para reconhecer os sinais de Libras

Para classificar a língua brasileira de sinais, diferentes modelos de aprendizagem de máquina têm sido utilizados na literatura. Citam-se como exemplos os trabalhos de Costa Filho et al. (2016 e 2017), Escobedo e Camara (2015 e 2016), Godoy et al. (2014), Porfirio et al. (2013) e Almeida, Guimaraes e Ramirez (2014) listados na Tabela 1.

O trabalho de Costa Filho et al. (2017) utiliza o classificador k-vizinhos mais próximos (*k-Nearest Neighbors* (kNN)) para classificar o conjunto de dados descrito em Costa Filho et al. (2016). A técnica de extração de características utilizada foi a $(2D)^2PCA$ (*Two-Directional Two-Dimensional Principal Component Analysis*), com tamanho do vetor de entrada $[100, 1]$, usando distância euclidiana e $k = 1$. O conjunto de dados foi dividido em 50% para treinamento e 50% para teste com a técnica *interleave*. A maior acurácia alcançada foi de 96,31%.

Para classificar as 61 configurações de mão adquiridas em imagens de profundidade através do sensor *Kinect*, o trabalho de Costa Filho et al. (2016) utiliza o classificador de novidades. Para extrair as características é utilizada a técnica $(2D)^2PCA$. Na construção do conjunto de dados, cada sinal foi realizado 20 vezes, por dez pessoas, totalizando 12.200 imagens de profundidade. A divisão dos dados foi realizada pela técnica *interleave* com 50% para treinamento e 50% para teste. A maior acurácia alcançada foi de 95,41%.

O trabalho de Escobedo e Camara (2016) utiliza máquina de vetores de suporte (*Support Vector Machine* (SVM)) com núcleo linear para classificar o conjunto de dados descrito em Escobedo e Camara (2015). Na extração de características, foram combinadas características globais e locais. Os dados foram divididos em 60% para treinamento e 40% para teste. A maior acurácia alcançada foi de 99,84%.

Para classificar 18 gestos da mão no formato de vídeo RGB (*Red Green Blue*), de profundidade e de esqueleto, capturados pelo *Kinect*, o trabalho de Escobedo e Camara (2015) utiliza máquinas de vetores de suporte com núcleo de função de base radial (*Radial Basis Function* (RBF)). Na extração de características, foram combinadas características globais e locais. Na construção do conjunto de dados os gestos foram realizados 20 vezes por duas pessoas. O

número de 10 quadros usados para cada vídeo foi estabelecido experimentalmente. A divisão dos dados em treinamento e teste não é citada. A melhor acurácia obtida foi de 98,28%.

Na construção do conjunto de dados, Godoy et al. (2014) capturam nove gestos no formato de vídeo RGB, de profundidade e de esqueleto, gravados pelo sensor *Kinect*. Cada gesto foi repetido pelo menos três vezes por 23 indivíduos. O total de amostras capturadas foi de 181, com cada amostra composta por 30 quadros. Nesse trabalho, não há citação a respeito da divisão dos dados em conjuntos de treinamento e de teste. A melhor acurácia alcançada foi de 91,2%, com um conjunto de classificadores de modelo de Markov oculto (*Hidden Markov Model* (HMM)), extração de características baseada na posição e trajetória da mão e com ampliação da quantidade de amostras através da técnica LOOT (*Leave One Out Training and Testing*).

O trabalho de Porfirio et al. (2013) utiliza máquinas de vetores de suporte com núcleo linear para classificar 61 configurações de mão gravados com o sensor *Kinect* no formato de vídeo RGB, de profundidade e de esqueleto. A extração de características foi realizada com o método de harmônicos esféricos. Na construção do conjunto de dados, as configurações de mão foram realizadas duas vezes por cinco indivíduos, totalizando 610 vídeos. Os quadros com vista frontal e lateral da mão foram extraídos manualmente e reconstruídos com o método *Silhouette*. Os dados foram separados randomicamente em 70% para treinamento e 30% para teste. A maior acurácia alcançada foi de 86,06%.

Para classificar 34 sinais obtidos através do sensor *Kinect* no formato de vídeo RGB, de profundidade e de esqueleto, o trabalho de Almeida, Guimaraes e Ramirez (2014) utiliza máquinas de vetores de suporte com núcleo RBF. As características foram extraídas com base na estrutura morfológica da língua brasileira de sinais. Para cada característica foi usada uma técnica diferente. Na construção do conjunto de dados, os sinais foram realizados 5 vezes por um único indivíduo, totalizando 170 vídeos. Para reduzir a quantidade de quadros presentes em cada vídeo é utilizada a técnica MDP (*Maximum Diversity Problem*). Os dados foram separados randomicamente em 60% para treinamento e 40% para teste. A taxa de acurácia para cada sinal fica acima de 80%.

Tabela 1 – Resumo dos artigos abordados na revisão bibliográfica para aprendizagem tradicional na classificação de Libras.

Referência	Materiais		Classificador	Acurácia
	Imagem/Vídeo	Quantidade		
Costa Filho et al. (2017)	**Imagem:** Profundidade	12.200	kNN	96,31%
Costa Filho et al. (2016)	**Imagem:** Profundidade	12.200	Novidades	95,41%
Escobedo e Camara (2016)	**Vídeo:** RGB, Profundidade e Esqueleto	Não informada	SVM	99,84%
Escobedo e Camara (2015)	**Vídeo:** RGB, Profundidade e Esqueleto	Não informada	SVM	98,28%
Godoy et al. (2014)	**Vídeo:** RGB, Profundidade e Esqueleto	181	HMM	91,2%
Porfirio et al. (2013)	**Vídeo:** RGB, Profundidade e Esqueleto	610	SVM	86,06%
Almeida, Guimaraes e Ramirez (2014)	**Vídeo:** RGB, Profundidade e Esqueleto	170	SVM	Acima de 80% por sinal

2.2.2 Aprendizagem profunda para reconhecer os sinais de Libras

Um primeiro esforço para reconhecer a língua brasileira de sinais (Libras) utilizando arquiteturas de redes neurais convolucionais está descrito no trabalho de Stiehl et al. (2015). Neste trabalho, os sinais de Libras são representados por símbolos pertencentes ao sistema de escrita *SignWriting*.

Para classificar os 103 símbolos do sistema de escrita *SignWriting* relacionados aos sinais pertencentes a língua brasileira de sinais, Stiehl et al. (2015) utilizam uma rede neural convolucional cuja arquitetura foi descrita previamente no trabalho de Ciresan, Meier e Schmidhuber (2012). Na construção do conjunto de dados, os símbolos foram fornecidos por trinta pessoas, surdas e não surdas, totalizando 7.997 imagens em escala de cinza.

A quantidade de dados utilizada para treinamento e teste não é citada. A maior acurácia alcançada foi de 94,4%, com o aumento dos dados através de cortes nas imagens em diferentes escalas, rotação em diferentes graus, suavização através do filtro de média e operações morfológicas de abertura e fechamento.

2.2.3 Apredizagem profunda para reconhecer outras línguas de sinais

As redes neurais convolucionais têm sido utilizadas na classificação da língua americana de sinais conforme mostram os trabalhos de Kang, Tripathi e Nguyen (2015), Li et al. (2015), Zhu et al. (2016) e Ameen e Vadera (2017) e na classificação da língua italiana conforme mostra o trabalho de Pigou et al. (2015). O trabalho de Huang et al. (2015) não faz menção a língua de sinais adotada.

Para classificar 31 sinais de mão pertencentes à língua americana de sinais (*American Sign Language* (ASL)), o trabalho de Kang, Tripathi e Nguyen (2015) utiliza a arquitetura *Caffenet* de rede neural convolucional existente na ferramenta *Caffe*. Na construção do conjunto de dados, os sinais foram capturados por um sensor *Creative Senz3D* e foram realizados 200 vezes por cinco indivíduos, totalizando 31.000 imagens de profundidade. A divisão dos dados é realizada da seguinte forma: quatro indivíduos para treinamento e um para teste, com variação entre os indivíduos até que todos tenham participado do treinamento e do teste. A maior acurácia alcançada foi de 85,49%, utilizando transferência de aprendizagem.

Para classificar o conjunto de dados descrito no trabalho de Pugeault e Bowden (2011), o trabalho de Li et al. (2015) propõe uma arquitetura de rede neural convolucional. As 24 letras do alfabeto, nos formatos RGB e profundidade, pertencentes à língua americana foram realizados por cinco indivíduos e capturados por um sensor *Kinect*, totalizando 120.000 imagens. Os dados foram divididos em 60.000 para treinamento e 60.000 para teste. A maior acurácia alcançada foi de 99,10% sem nenhuma técnica de regularização.

O trabalho de Zhu et al. (2016) propõe uma arquitetura de rede neural convolucional para classificar o conjunto de dados descrito no trabalho de Pugeault e Bowden (2011). Os dados foram divididos em 100.000 para treinamento e 20.000 para teste. A maior acurácia alcançada

foi de 88,3%, sem nenhuma técnica de regularização.

O trabalho de Ameen e Vadera (2017) propõe uma arquitetura de rede neural convolucional para classificar o conjunto de dados descrito no trabalho de Pugeault e Bowden (2011). Os dados foram divididos por indivíduo: quatro para treinamento e um para teste. Os indivíduos foram variados até que todos tivessem participado do treinamento e do teste. O resultado final é a média desses resultados parciais. A maior acurácia alcançada foi de 80,34% sem nenhuma técnica de regularização.

Pigou et al. (2015) propõe uma arquitetura de rede neural convolucional para classificar o conjunto de dados do desafio *Chalearn Looking at People 2014*, com 6600 gestos da língua italiana executados por 27 indivíduos e capturados pelo sensor *Kinect* (ESCALERA et al., 2014). Os dados foram divididos em 4600 para treinamento e 2000 para validação. A maior acurácia alcançada foi de 91,70% utilizando *dropout*, LCN (*Local Contrast Normalization*) e aumento de dados.

Huang et al. (2015) propõe uma arquitetura de rede neural convolucional para classificar 24 gestos existentes em vídeo RGB, de profundidade e de esqueleto. Na construção do conjunto de dados, os gestos foram realizados 3 vezes por 9 indivíduos, totalizando 675 vídeos capturados pelo sensor *Kinect*. A língua a qual os gestos pertencem não é citada. Os dados foram divididos em 450 para treinamento e 225 para teste. A acurácia obtida foi de 94,2%, sem nenhuma técnica de regularização.

A Tabela 2 apresenta um resumo de todos os trabalhos citados nesta seção.

Tabela 2 – Resumo dos artigos abordados na revisão bibliográfica para aprendizagem profunda na classificação de outras línguas de sinais.

Referência	**Materiais**			**Acurácia**
	Língua de Sinais	**Imagem/Vídeo**	**Quantidade**	
Kang, Tripathi e Nguyen (2015)	ASL	**Imagem:** Escala de cinza e Profundidade	31.000	85,49%
Li et al. (2015)	ASL	**Imagem:** RGB e Profundidade	120.000	99,1%
Zhu et al. (2016)	ASL	**Imagem:** RGB e Profundidade	120.000	88,3%
Ameen e Vadera (2017)	ASL	**Imagem:** RGB e Profundidade	120.000	80,34%
Pigou et al. (2015)	Italiana	**Vídeo:** Escala de cinza, Profundidade, Esqueleto e Índice	6.600	91,7%
Huang et al. (2015)	Não informada	**Vídeo:** RGB, Profundidade e Esqueleto	675	94,2%

2.3 Conclusão da revisão bibliográfica

Os trabalhos revisados na seção 2.2.1, mostram que existem pesquisas realizadas para classificar a língua brasileira, mas que as técnicas de classificação adotadas ainda são as que exigem uma etapa de pré-processamento de imagem para extração de características. Além disso, é possível notar, pelo levantamento realizado, que há uma tendência em realizar pesquisas nessa área utilizando vídeos. Em contrapartida, os conjuntos de dados apresentados nos trabalhos são proprietários. Nesses casos, é comum os proprietários disponibilizarem para outros pesquisadores somente após vasta exploração dos mesmos.

A identificação de apenas um trabalho revisado na seção 2.2.2, demonstra que há um nicho pouco explorado com relação à utilização de redes neurais convolucionais na classificação da língua brasileira de sinais, e esse trabalho, embora seja um precursor desse tema, trabalha com imagens do tipo esqueleto, ou seja, com menos informação do que as imagens existentes no conjunto de dados criado pelo Grupo de Pesquisa em Reconhecimento de Padrões e Otimização.

Os trabalhos revisados na seção 2.2.3, demonstram que há pesquisas abordando a utilização de redes neurais convolucionais com as línguas de sinais americana e italiana e, tal fato, motiva a investigação da viabilidade de reconhecimento da língua brasileira de sinais com a utilização de redes neurais convolucionais.

Com isso, a revisão realizada possibilitou que este trabalho integrasse:

- A utilização de redes neurais convolucionais, técnica de classificação usada em sete trabalhos revisados;
- O conjunto de dados *LibrasImage*, criado pelo Grupo de Pesquisa em Reconhecimento de Padrões e Otimização e disponível para utilização;
- Um artigo de referência, o trabalho de Costa Filho et al. (2017) para o *benchmark*.

3 FUNDAMENTAÇÃO TEÓRICA

Para compreender a utilização de redes neurais convolucionais em tarefas classificação, foi necessário pesquisar referências que descrevessem esse tema. Neste capítulo, são descritos os conceitos relacionados a construção de modelos de classificação.

3.1 Redes neurais convolucionais

A aprendizagem de máquina é uma técnica que gera um modelo a partir dos dados (GÉRON, 2017), sejam eles documentos, áudios, vídeos ou imagens. A técnica consiste em analisar os dados e encontrar o modelo sem que este precise previamente ser informado por um ser humano.

Em aprendizagem de máquina, a entrada x e a saída y são usadas para criar uma função que relaciona essas duas variáveis. A essa função denomina-se função de previsão $h(x)$, que é a função que se acredita ser a mais próxima da função verdadeira $f(x)$ (GOODFELLOW; BENGIO; COURVILLE, 2016). O processo usado para definir a função de previsão é chamado de treinamento do modelo. Quando a função de previsão é definida, é possível prever valores de y correspondentes a x.

Se a previsão do valor y for diferente do valor conhecido, diz-se que há um erro de previsão. A soma dos erros sobre todo o conjunto de treinamento é uma boa forma de mensurar a acurácia do modelo. Entre as diferentes técnicas de aprendizagem de máquina estão: a aprendizagem supervisionada, a aprendizagem não supervisionada e a aprendizagem por reforço (LI; J.; S., 2017). Na aprendizagem supervisionada, é preciso conhecer a solução de cada entrada para que esta possa ser comparada com o resultado informado pelo modelo.

A primeira meta de qualquer algoritmo de aprendizagem supervisionada é minimizar o erro enquanto a função de previsão $h(x)$ está sendo definida na etapa de treinamento do modelo. Entre as tarefas que podem ser executadas em aprendizados supervisionados temos a regressão e classificação (LI; J.; S., 2017). A classificação consiste em separar os dados em categorias ou classes discretas. Após o modelo ter sido treinado, novos dados devem ser apresentados em sua entrada, para que o mesmo realize as devidas classificações. Quando um modelo realiza as classificações para as novas entradas corretamente, diz-se que ele tem uma boa capacidade de generalização.

Uma das formas de obter os modelos de aprendizagem de máquina é através das redes neurais artificiais, que imitam o mecanismo do cérebro humano, que é composto de numerosos neurônios. Os neurônios são representados nas redes neurais por nós, que são associados através de pesos. A partir de como os nós estão conectados é possível criar uma variedade de redes neurais.

A arquitetura mais simples de uma rede neural é composta por uma camada de entrada e uma de saída, denominada de rede neural de uma única camada (*single-layer neural network*).

Quando uma ou mais camadas escondidas são adicionadas em uma rede neural de um única camada, são produzidas as redes neurais multicamadas. Uma rede neural multicamada composta por uma única camada escondida é denominada de rede neural *shallow* ou rede neural *vanilla*. Quando uma rede neural possui duas ou mais camadas escondidas, é chamada de rede neural profunda (*Deep Neural Network* (DNN)) (PATTERSON; GIBSON, 2017).

Entre as principais arquiteturas de redes neurais profundas, temos as redes neurais convolucionais (*Convolutional Neural Networks* (CNN)) (KUMAR; DESHPANDE, 2018). As redes neurais convolucionais são tipicamente utilizada para classificar objetos em imagens. Em pelo menos uma de suas camadas, as mesmas empregam uma operação matemática chamada de convolução. Nessa operação, as informações, dos dados de entrada e do filtro ou núcleo (*kernel*), são multiplicadas e somadas para produzir a saída convoluída, conforme ilustra a Figura 1.

A convolução melhora o aprendizado de uma máquina devido a três características: "iterações esparsas", "compartilhamento de parâmetros"e "representações equivariantes"(KIM, 2017). O conceito de iteração esparsa faz com que cada unidade de uma camada precedente sirva de ativação para apenas algumas unidades da camada seguinte. Esse tipo de processamento é rápido, quando comparado a multiplicação de matrizes nas redes neurais tradicionais.

O conceito de compartilhamento de parâmetros permite que um conjunto de pesos seja aplicado em operações de convolução em diferentes regiões, normalmente do tamanho de um núcleo, de uma camada precedente, para gerar ativações da camada seguinte.

O conceito de representação equivariante, no processamento de imagens através da convolução de um núcleo, permite que os resultados sejam invariantes em relação a operações de translação. Porém, a convolução não é equivariante a transformações de escala e rotação de uma imagem.

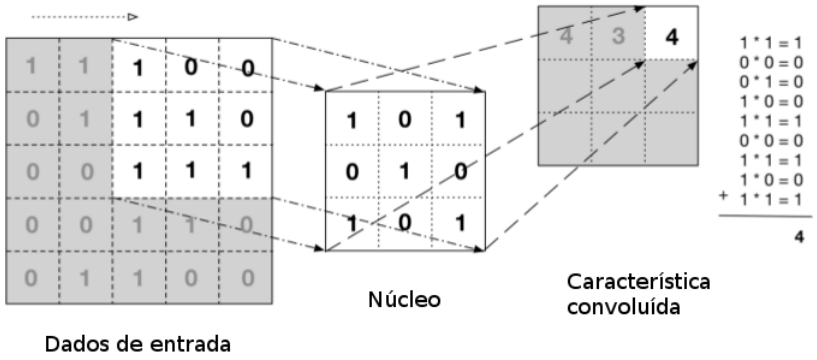


Figura 1 – Ilustração da operação de convolução realizada pelas redes neurais convolucionais. A máscara percorre os dados de entrada da esquerda para a direita e de cima para baixo. Os valores da máscara são multiplicados pelos valores dos dados de entrada e somados. O resultado é a saída convoluída.

Fonte – Adaptada de Patterson e Gibson (2017).

Assim, as redes neurais convolucionais são um tipo de arquitetura especializada na classificação de imagens, que utilizam a operação de convolução para obter um processamento mais rápido e extrair características semelhantes em áreas diferentes de uma mesma imagem.

Diferentes camadas de convolução aplicadas a uma mesma imagem permitem extrair diferentes características da mesma.

Uma rede neural convolucional usada para classificação é composta por múltiplas camadas que exercem funções distintas (KIM, 2017; PATTERSON; GIBSON, 2017; GOODFELLOW; BENGIO; COURVILLE, 2016; TOK; DEAN; M., 2018; GÉRON, 2018; MENSHAWY, 2018; PATTANAYAK, 2017).

A primeira camada de uma arquitetura de rede neural convolucional é a camada de entrada, que determina a largura e altura da imagem que será lida, bem como a quantidade de canais que a imagem deve possuir. A camada de entrada apresentada na Figura 2, está configurada para receber uma imagem com 139 pixels de largura, 135 pixels de altura e 1 canal. Para imagens no formato RGB, a quantidade de canais deve estar configurada com o valor 3.

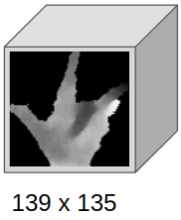


Figura 2 – Ilustração de uma camada de entrada. Largura da imagem: 139 pixels. Altura da imagem: 135 pixels. Quantidade de canais: 1.

A segunda camada de uma arquitetura de rede neural convolucional é a camada de convolução. A camada de convolução converte a imagem lida na camada de entrada em mapas de características. Essa conversão é realizada através de filtros de convolução, *stride* e *padding*.

Nas configurações do filtro são determinadas a quantidade de filtros que serão utilizados, bem como a altura e largura de cada filtro. O *stride* determina o número de pixels que serão utilizados para mover o filtro sobre a imagem. O *padding* determina o número de zeros que serão adicionados às bordas da imagem para controlar o tamanho da saída da convolução. A Figura 3 apresenta uma camada de convolução configurada com 96 filtros de tamanho 11 x 11 com *stride* de 4 e *padding* de 0 aplicados em uma imagem de entrada.

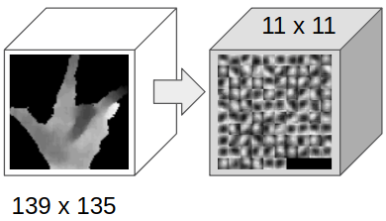


Figura 3 – Ilustração de uma camada de convolução com 96 Filtros de tamanho de 11 x 11 com um *stride* de 4 e *padding* de 0.

Após a camada de convolução são aplicadas as funções de ativação. Existem referências como a de Patterson e Gibson (2017), Goodfellow, Bengio e Courville (2016) que apresentam a função de ativação como uma camada adicional, separada da camada de convolução. A função de ativação geralmente aplicada em redes neurais convolucionais é a ReLU, que substitui todos

os pixels negativos por zero. A Figura 4 apresenta a função de ativação ReLU aplicada após uma camada de convolução.

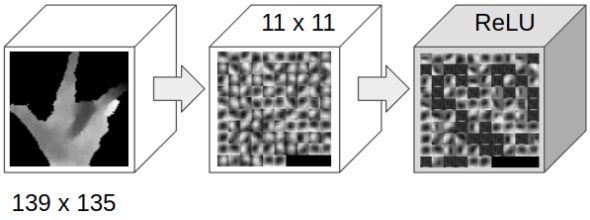


Figura 4 – Ilustração da função de ativação ReLU.

A terceira camada de uma arquitetura de rede neural convolucional é a camada de agrupamento. A camada de agrupamento proporciona a redução da dimensão das características aprendidas, com a combinação de pixels vizinhos de uma área de agrupamento definida por um filtro e um *stride*, em um valor definido pelo tipo de agrupamento que pode ser escolhido entre o de média ou máximo. As camadas de convolução, ReLU e agrupamento podem ser repetidas várias vezes. A Figura 5 apresenta uma camada de agrupamento de máximo configurada com filtros de tamanho 3 x 3 e *stride* de 2.

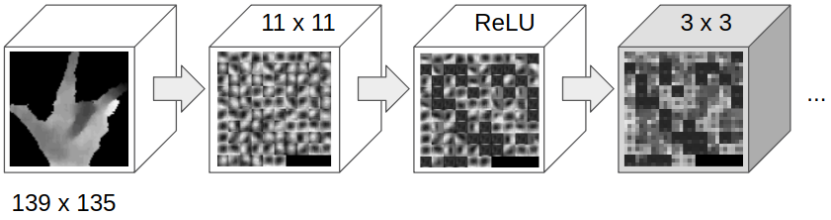


Figura 5 – Ilustração de uma camada de agrupamento com filtros de tamanho 3 x 3 com um *stride* de 2.

A próxima camada de uma arquitetura de rede neural convolucional é a camada totalmente conectada. Essa camada transforma o mapa de características em um vetor. Ela contém os neurônios que recebem diferentes conjunto de pesos da camada precedente. Cada neurônio nessa camada será conectado a todos os neurônios da camada anterior. A Figura 6 apresenta uma camada totalmente conectada configurada com 4096 neurônios.

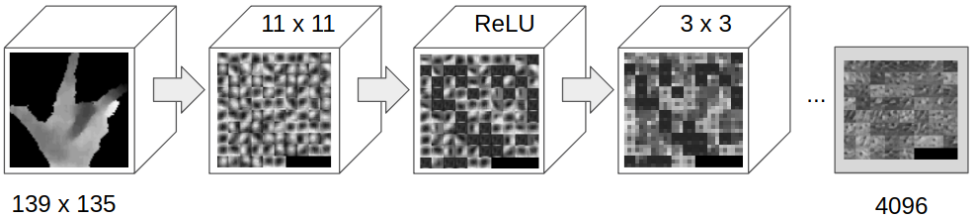


Figura 6 – Ilustração de uma camada totalmente conectada configurada com 4096 neurônios.

A última camada de uma arquitetura de rede neural convolucional é a camada de saída. Essa camada também é totalmente conectada e sua dimensão é igual a quantidade de classes que o conjunto de dados utilizado apresenta. A Figura 7 apresenta uma camada de saída com 61 classes existentes.

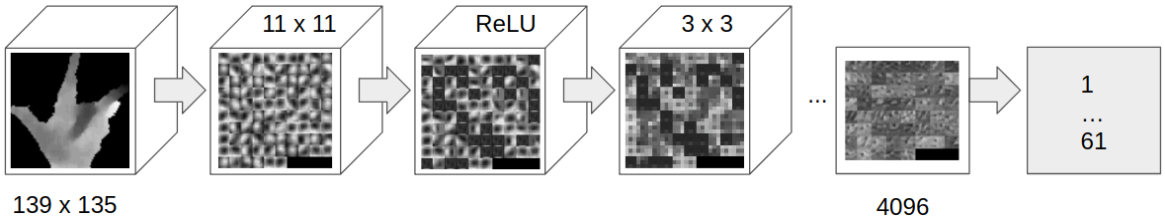


Figura 7 – Ilustração de uma camada de saída com 61 classes.

3.2 Algoritmo de aprendizagem

O gradiente descendente estocástico com momento (*Stochastic Gradient Descent with Momentum* (SGDM)) é um algoritmo de aprendizagem de primeira ordem usado para treinar redes neurais convolucionais. Os algoritmos de primeira ordem calculam a matriz Jacobiana, que é uma matriz composta de derivadas parciais da função de perda com relação aos parâmetros da rede (GÉRON, 2017; GOODFELLOW; BENGIO; COURVILLE, 2016).

O algoritmo dá um passo na direção da função Jacobiana (ou simplesmente Jacobiano), que é o determinante da matriz Jacobiana (BONACCORSO, 2017). Isso significa que a cada passo, o algoritmo está tentando encontrar a melhor direção a seguir, de acordo com a função objetivo estabelecida, ou seja, encontrar um caminho cujo erro seja mínimo.

Em vez de considerar todo o conjunto de dados disponibilizado para treinamento da rede neural convolucional, a atualização dos pesos é aplicada nos mini lotes extraídos do conjunto de treinamento. O processo realizado pelo algoritmo deve finalizar quando os pesos param de modificar ou quando suas variações se mantêm abaixo de um limiar selecionado.

Quando o algoritmo alcança esse estágio diz-se que ele atingiu um ponto ótimo (PATTERSON; GIBSON, 2018). Em cenários mais simples, o ponto ótimo tem grandes chances de ser o mínimo global, em cenários mais complexos, há um grande risco do ponto ótimo ser na verdade um ponto sub ótimo, e para este caso, estratégias mais complexas devem ser implementadas para que o máximo desempenho seja alcançado.

O processo de treinamento de uma rede neural convolucional acontece com a inicialização dos pesos de maneira aleatória e em seguida entrada no *loop* que permite sucessivas iterações. Em cada interação, é selecionado um mini lote do conjunto de treinamento, são definidos os valores esperados e calculados os valores reais para esse mini lote. Se a classe prevista não for igual a classe real para todas as amostras do mini lote, os pesos do vetor são atualizados usando o gradiente descendente estocástico com momento e o erro é contabilizado. O *loop* finaliza quando o erro contabilizado for muito próximo de zero, pois significa que o algoritmo convergiu para um ponto ótimo, ou de acordo com condições de parada definidas previamente.

O SGDM atualiza os parâmetros, pesos e polarizações conforme a Equação (1) (PATTERSON; GIBSON, 2017), onde l representa o número da iteração, θ é o parâmetro que será atualizado, α é a taxa de aprendizagem, $\nabla J(\theta)$ é o gradiente da função de perda e γ representa a

contribuição do gradiente anterior para a iteração atual, ou seja o momento.

$$\theta_{l+1} \leftarrow \theta_l - \alpha \nabla J(\theta) + \gamma(\theta_l - \theta_{l-1}) \tag{1}$$

3.3 Hiperparâmetros em nível de algoritmo de aprendizagem

Em aprendizagem de máquina, existem os parâmetros do modelo, calculados pelo algoritmo de aprendizagem, e os parâmetros que são configurados, para que as redes treinem melhor e mais rápido. Esses parâmetros configurados são chamados de hiperparâmetros (GOODFELLOW; BENGIO; COURVILLE, 2016). A seleção dos hiperparâmetros se baseia na verificação de que o modelo não sofre *underfitting* sobre o conjunto de dados de treinamento enquanto aprende a estrutura dos dados.

Os hiperparâmetros podem ser divididos em nível de estrutura e topologia de uma rede neural ou em nível de algoritmo de aprendizagem (GÉRON, 2017). Os hiperparâmetros em nível de estrutura e topologia de rede dizem respeito à quantidade de camadas, número de neurônios, funções de ativação, tamanho dos filtros, *stride* e *padding* etc. Os hiperparâmetros em nível de algoritmo de aprendizagem consistem em variáveis externas ao modelo cujo valor não pode ser estimado a partir dos dados, como:

- Taxa de aprendizagem;
- Tamanho do mini lote;
- Momento;
- Decaimento da taxa de aprendizagem;
- Número de épocas.

A taxa de aprendizagem pode ser considerada o mais importante hiperparâmetro (PATTERSON; GIBSON, 2017), porque controla a capacidade do modelo tentando minimizar a função de perda. Encontrar uma boa taxa de aprendizagem pode ser complicado, porque uma taxa de aprendizagem grande pode aumentar o erro de treinamento e fazê-lo ter problemas para convergir para um mínimo global, enquanto que uma taxa de aprendizagem pequena pode tornar o treinamento lento e fazê-lo convergir para um mínimo local ao invés do mínimo global, conforme ilustra a Figura 8.

Patterson e Gibson (2017) recomendam iniciar com um valor de taxa grande e, se o modelo divergir, esse valor deve ser dividido por um fator. Esse procedimento deve ser repetido até a convergência. Eles sugerem iniciar os treinamentos com os seguintes valores de taxa: 0,1; 0,01 e 0,001. Para encontrar uma boa taxa de aprendizagem é preciso realizar o treinamento da rede e gerar a curva de aprendizagem para cada um dos valores que se deseja verificar. Segundo os autores, a taxa de aprendizagem ideal aprenderá rapidamente e convergirá para uma boa solução.

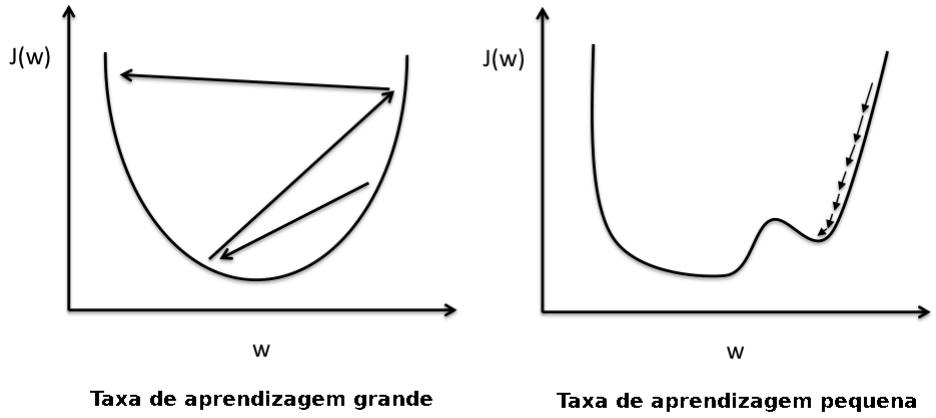


Figura 8 – Ilustração de escolhas inadequadas de taxa de aprendizagem.

Fonte – Adaptada de Araujo et al. (2018).

O tamanho do mini lote é um hiperparâmetro que determina o particionamento dos dados de treinamento em mini lotes que passarão pela rede. Cada mini lote é carregado e alimentado na rede onde o algoritmo do gradiente descendente estocástico calcula os pesos e os atualiza (TORRES, 2018). O gradiente calculado para os mini lotes são uma estimativa do verdadeiro gradiente para todo o conjunto de treinamento, mas essas pequenas atualizações repetidamente, levam a uma convergência próxima o suficiente de um bom mínimo para a função de perda. Isso é repetido até que o conjunto de dados de treinamento tenha completamente passado pela rede. Nesse momento, diz-se que o modelo completou uma época.

Por exemplo, para um conjunto de treinamento com 6100 imagens e um mini lote de tamanho 128, uma época equivale a 47,66 atualizações do modelo. O tamanho dessa variável depende da capacidade de memória do computador que será usado para realizar os cálculos. Mini lotes grandes dão uma estimativa melhor do verdadeiro gradiente e permitem o uso de taxas de aprendizagem maiores, em contrapartida exigem mais memória durante o treinamento.

O momento é um hiperparâmetro que pode fazer com que o processo de aprendizagem mova os parâmetros em uma direção de maneira menos abrupta (ARAUJO et al., 2018). Dessa forma, determinadas regiões do espaço de pesquisa podem ser exploradas sem que o processo de aprendizagem sinta o efeito total das variações existentes. Com isso, as mudanças em direção à aprendizagem ao longo do tempo ficam mais suaves e se evita oscilações muito grandes. Isso acontece porque o momento é uma constante que pondera os gradientes anteriores.

O decaimento da taxa de aprendizagem é um hiperparâmetro que ajuda a diminuir a taxa de aprendizagem à medida que as épocas vão passando, para que a aprendizagem aconteça de forma mais rápida (BUDUMA; LOCASCIO, 2017). Na fase inicial da aprendizagem esse ajuste é maior e à medida que as épocas vão passando esse ajuste fica cada vez menor para facilitar que o treinamento convirja ao mínimo da função de perda.

O número de épocas é um hiperparâmetro que indica a quantidade de iterações sobre todo o conjunto de treinamento que passará pela rede. Por exemplo, ao definir o número de épocas como sendo 500, todo o conjunto de treinamento passará pela rede 500 vezes.

3.4 Técnicas de regularização

Um conjunto de dados disponibilizado para a etapa de treinamento de uma rede neural convolucional é normalmente uma representação que contém apenas alguns elementos de um conjunto de dados total. A etapa de treinamento consiste em encontrar um modelo apropriado que mantenha a capacidade de generalizar quando uma entrada não conhecida for apresentada.

A capacidade de representação de um modelo é considerada baixa (*underfitting*), quando um modelo não tem habilidade de capturar os detalhes contidos no conjunto de treinamento, ou alta (*overfitting*), quando um modelo captura bem os detalhes contidos no conjunto de treinamento, mas quando uma entrada não conhecida é apresentada, o modelo classifica de maneira errada (GÉRON, 2017).

Um modelo com capacidade baixa geralmente tem um alto *bias*, que é a diferença entre o valor esperado, $E[\theta]$, e o valor real, θ, dos parâmetros (HOPE; RESHEFF; LIEDER, 2017), conforme apresenta a Equação (2). A presença de um alto *bias* mostra que o algoritmo não está aprendendo uma representação aceitável do parâmetro. Na fase de treinamento é mais fácil identificar um modelo com capacidade baixa, pois o mesmo apresenta erros durante a execução.

$$Bias[\theta] = E[\theta] - \theta \tag{2}$$

Um modelo com capacidade alta normalmente tem uma alta variância, que é a esperança da diferença entre o valor real e esperado dos parâmetros elevada ao quadrado (GOODFELLOW; BENGIO; COURVILLE, 2016), conforme ilustra a Equação (3). Na fase de treinamento é mais difícil identificar um modelo com capacidade alta, pois nesta fase, o modelo apresenta um ajuste muito próximo do conjunto de dados treinado.

$$Variância[\theta] = E[(\theta - E[\theta])^2] \tag{3}$$

Para evitar problemas de *overfitting*, algumas técnicas de regularização podem ser usadas como:

- L2;
- *Dropout*.

A técnica de regularização L2 ou comumente chamada de decaimento de peso (*weight decay*) deve ser usada em modelos densos e quando não se está usando parada antecipada (PATTERSON; GIBSON, 2017). Essa técnica evita que os pesos da conexão da rede se tornem muito grandes através da adição de um termo de regularização $\lambda\Omega(\mathbf{w})$ na função objetivo $E(\theta)$.

A função objetivo regularizada ($E_R(\theta)$) assume a forma da Equação (4), em que λ é um coeficiente de regularização, $\mathbf{w}$ é o vetor de pesos e $\Omega(\mathbf{w})$ diminui pela metade o comprimento dos pesos da rede, calculado pela norma dos mesmos, conforme mostra a Equação (5).

$$E_R(\theta) = E(\theta) + \lambda\Omega(\mathbf{w}) \tag{4}$$

$$\Omega(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \tag{5}$$

A técnica de regularização *dropout* consiste em excluir temporariamente a cada passo de treinamento um determinado neurônio que será ativado no próximo passo (BONACCORSO, 2017), conforme ilustra a Figura 9. Com isso, a técnica treina todas as sub-redes que podem ser formadas. Ela funciona bem com o gradiente descendente estocástico. É mais efetiva se aplicada a todas as camadas escondidas e menos efetiva se o conjunto de treinamento aumenta em dezenas de milhões. É frequentemente combinada com a técnica L2.

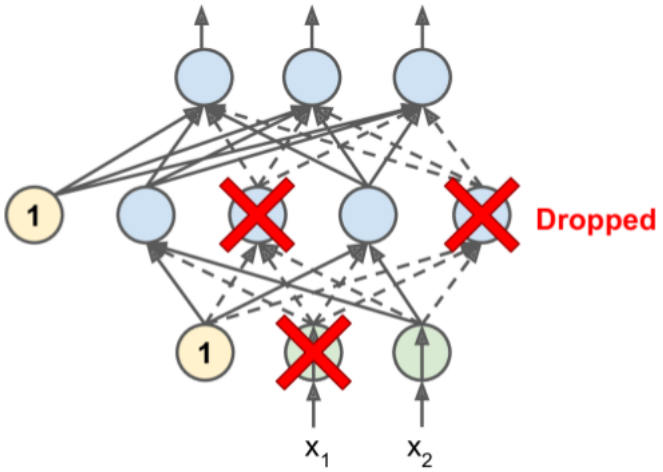


Figura 9 – Ilustração do processamento realizado na técnica de *dropout*.

Fonte – Géron (2017).

4 MATERIAIS E MÉTODOS

Para avaliar se a utilização de redes neurais convolucionais é capaz de melhorar a classificação das configurações de mão da língua brasileira de sinais, é necessário treinar modelos com essa técnica de classificação. Neste capítulo, são descritos os materiais utilizados para treinamento dos modelos e os métodos realizados para que a melhor capacidade de generalização fosse alcançada.

4.1 Materiais

4.1.1 Conjunto de dados

O conjunto de dados utilizado no trabalho de Costa Filho et al. (2017) é proprietário, ou seja, foi um conjunto de dados criado pelos autores e foi cedido para realização deste trabalho. Os autores disponibilizaram tanto o conjunto de dados original, com a imagem de profundidade do indivíduo realizando a configuração de mão (formato que a mão apresenta quando alguém realiza um sinal), quanto o pré-processado, com a imagem de profundidade contendo apenas a configuração de mão.

O conjunto de dados é composto por 61 configurações de mãos pertencentes à língua brasileira, adquiridos através de um sensor *Kinect*® e armazenados em imagens de profundidade de 640x480 pixels com 11 bits de resolução. Cada configuração de mão foi realizada 20 vezes por dez voluntários, totalizando 200 imagens por configuração. O conjunto de dados completo é composto de 12.200 imagens.

A captura das configurações foi realizada enquanto os indivíduos moviam a mão. As configurações capturadas foram coletadas em diferentes posições relativas ao corpo e diferentes rotações da mão, variando de 45° a 135°. A Figura 10 ilustra essa característica para a configuração de mão 36.

Figura 10 – Ilustração da captura realizada para a configuração de mão 36 em diferentes posições e inclinações.

A partir da consideração realizada pelos autores de que a mão era o objeto mais próximo

da câmera, foram encontrados os componentes conectados da região mais próxima da câmera. Como a região incluía parte do antebraço, a próxima etapa foi removê-lo. Por fim, as imagens foram redimensionadas para um tamanho único de 135x139 pixels, através de interpolação bilinear. O conjunto de dados pré-processado, resultante dessas etapas, está ilustrado na Figura 11, e é o conjunto de dados escolhido para ser utilizado neste trabalho. O mesmo conjunto de dados também foi utilizado no trabalho de referência.

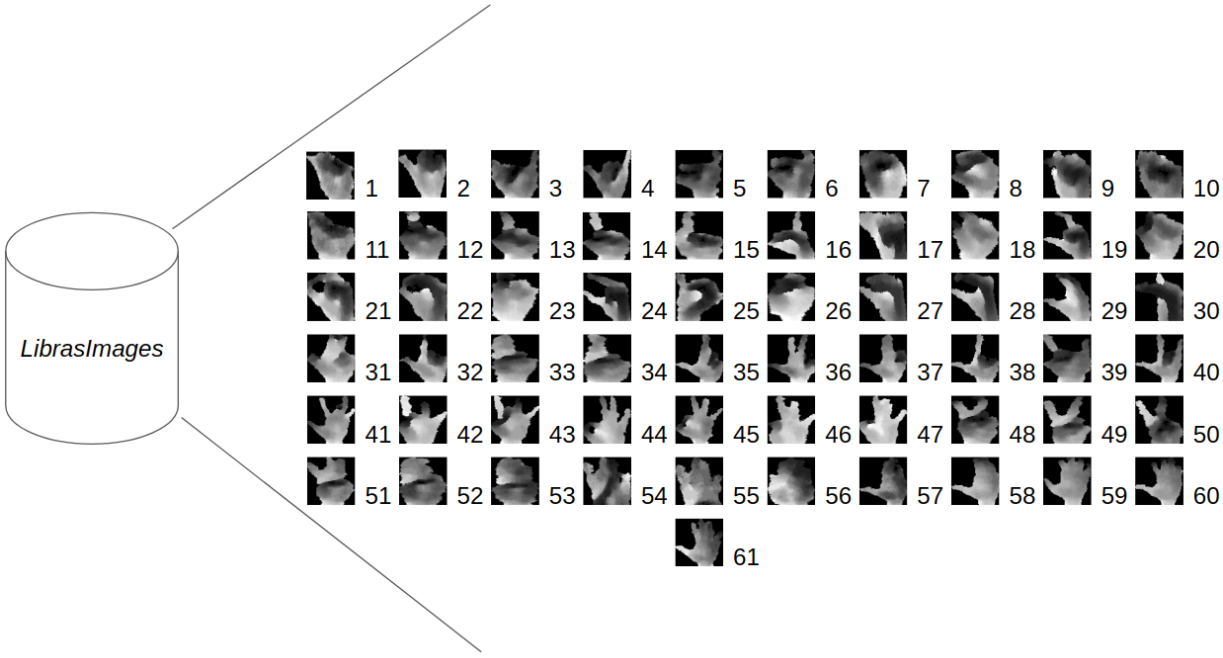


Figura 11 – Ilustração das 61 configurações de mão pertencentes ao conjunto de dados *LibrasImage* realizadas por um mesmo indivíduo.

4.1.2 Ambiente utilizado para implementação

Este trabalho foi realizado no Laboratório de Reconhecimento de Padrões e Otimização (PR&I Lab), do Centro de Tecnologia Eletrônica e da Informação (CETELI) da Universidade Federal do Amazonas (UFAM).

Os dois computadores utilizados são apresentados na Figura 12 e possuem as seguintes características:

- Marca: XPS;
- Modelo: Dell;
- Memória RAM: 16 GB;
- Processador: Intel(R) Core (TM) i7-4790 CPU @ 3.60GHz;
- Sistema operacional: Windows 8.1.

- Unidade gráfica: NVIDIA;
- Modelo da unidade gráfica: GeForce GTX 745;
- Memória RAM da unidade gráfica: 4 GB, com 384 núcleos;
- Versão do driver da unidade gráfica: 390.77.

Figura 12 – Ilustração do ambiente utilizado para treinamento e teste das redes neurais convolucionais.

O *software* utilizado para implementação das redes neurais convolucionais foi a *toolbox* de aprendizagem profunda disponível no *Matlab*, que permite realizar tarefas de classificação de imagens e possibilita monitoramento do treinamento de forma gráfica em tempo de execução (MATLAB, 2017). A versão do *Matlab* utilizada foi a R2017b.

No *Matlab*, a arquitetura da rede pode ser composta pelas seguintes camadas: entrada, convolução, normalização de lotes, ReLU, normalização de resposta local, agrupamento de máximo ou média, *dropout*, completamente conectada e saída.

Os parâmetros referentes ao momento, taxa de aprendizagem, coeficiente da regularização L2, quantidade de épocas, validação, modo CPU ou GPU, ponto de checagem e exibição de gráficos em tempo de execução são definidos nas opções de treinamento. No final do treinamento, o modelo é salvo em um arquivo binário de extensão *mat*. Para evitar perder um período de treinamento em situações adversas, como falta de energia, é possível salvar um modelo a cada época. Para armazenamento de imagens, o *Matlab* usa o *imageDatastore*.

A Figura 13 apresenta uma arquitetura de rede neural convolucional criada no Matlab, com uma camada de entrada, seguida de duas camada de convolução, ReLU e agrupamento de máximo, e duas camadas completamente conectadas.

As opções de treinamento aparecem logo abaixo da arquitetura com taxa de aprendizagem de 0,001, coeficiente de regularização desabilitado, quantidade de épocas para executar o treinamento definida em 500, opção para mostrar as informações do treinamento habilitada, opção para exibir graficamente o progresso de treinamento habilitada e definição do caminho para salvar um modelo a cada época do treinamento. A função de treinamento é chamada logo abaixo, com as informações referentes às imagens de entrada, arquitetura da rede e parâmetros de treinamento.

```
layers = [
    imageInputLayer( [ 135 139 1 ] )

    convolution2dLayer(5,16)
    reluLayer
    maxPooling2dLayer(3,'Stride',2)

    convolution2dLayer(5,32)
    reluLayer
    maxPooling2dLayer(3,'Stride',2)

    fullyConnectedLayer(512)
    reluLayer

    fullyConnectedLayer(61)
    softmaxLayer
    classificationLayer
];

options = trainingOptions('sgdm',...'
    'InitialLearnRate', 0.001, ...
    'L2Regularization', 0, ...
    'MaxEpochs', 500,...
    'Verbose', true,...
    'Plots',' training-progress', ...
    'CheckpointPath','checkpoint\');

net = trainNetwork(trainSet, layers, options);
```

Figura 13 – Ilustração de uma arquitetura de rede neural convolucional implementada no *Matlab*.

4.2 Métodos

Para alcançar os objetivos geral e específicos deste trabalho, foram realizados as etapas apresentadas no diagrama de blocos da Figura 14. Cada um dos métodos presentes nesse diagrama, estão descritos nas próximas seções.

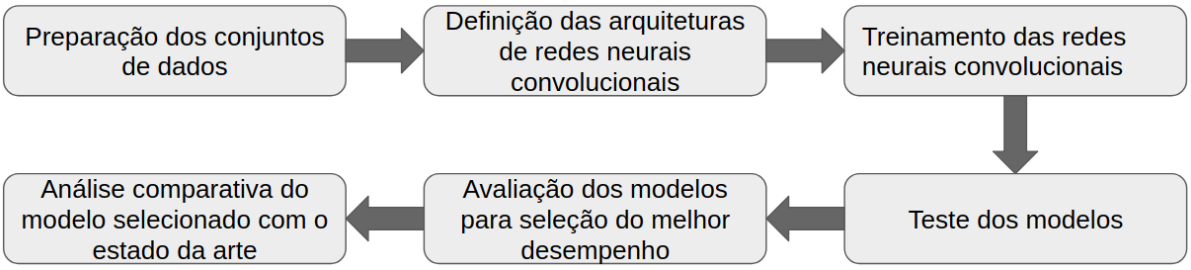


Figura 14 – Diagrama de blocos que apresenta os métodos realizados nesta pesquisa.

4.2.1 Preparação dos conjuntos de dados

O conjunto de dados original com 12.200 imagens foi dividido em conjuntos de treinamento e teste, com as seguintes porcentagens: 50% para treinamento e 50% para teste. Para que os dados de todos os indivíduos estivessem presentes no conjunto de treinamento e teste, a técnica utilizada para realizar essa divisão foi a *interleave*. Na técnica *interleave*, uma imagem é

colocada no conjunto de treinamento, enquanto que a próxima imagem é colocada no conjunto de teste. Esse procedimento é repetido até que não haja mais imagens. (vide Figura 15).

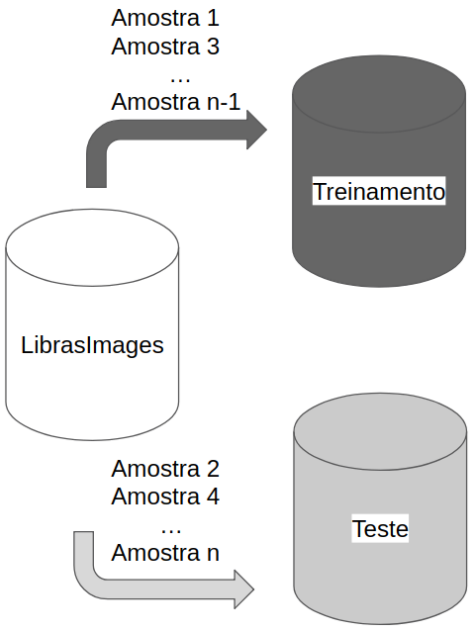


Figura 15 – Ilustração da técnica *interleave* utilizada na divisão do conjunto de dados.

Para armazenar as imagens dos conjuntos de treinamento e teste, foi usada a estrutura de dados do Matlab *imageDatastore*. Essa estrutura de dados é um objeto que gerencia uma coleção de arquivos de imagens (MATLAB, 2017). Com o *imageDatastore* o gerenciamento das imagens pode ser realizado através da própria função, sem manipulações manuais para leitura dos arquivos ou criação dos rótulos.

Para criar um objeto *imageDatastore*, é necessário usar a função que vem disponível no *Matlab*. Neste trabalho, as propriedades utilizadas para essa função foram:

- Incluir subpastas: verdadeiro;
- Fonte dos rótulos: nome das subpastas.

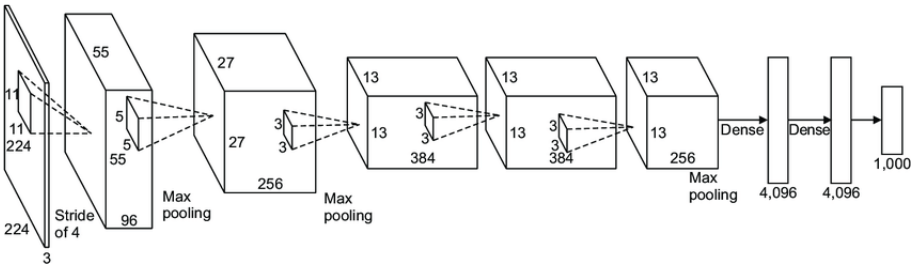
4.2.2 Definição das arquiteturas de redes neurais convolucionais

As arquiteturas foram selecionadas com base na revisão bibliográfica apresentada no capítulo 2. A escolha foi baseada nos seguintes critérios: tipo de dados (imagem) utilizado, utilização de aprendizagem supervisionada e melhor desempenho. Com base nos critérios estabelecidos, foram selecionadas as arquiteturas apresentadas nos trabalhos de Kang, Tripathi e Nguyen (2015), Zhu et al. (2016) e Pigou et al. (2015).

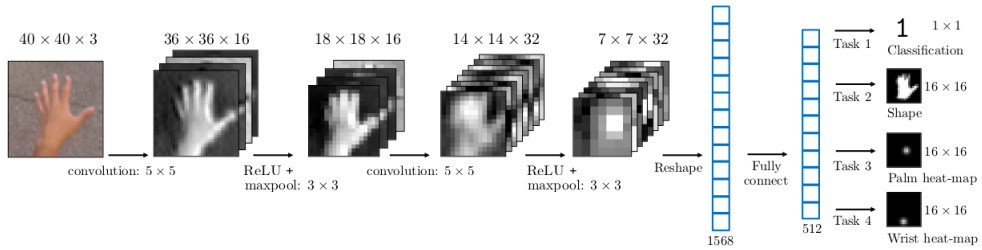
Kang, Tripathi e Nguyen (2015) utilizaram a arquitetura *Caffenet*, apresentada na Figura 16 (a), com imagem de profundidade em escala de cinza. A arquitetura *Caffenet* é uma adaptação da arquitetura *Alexnet* criada para o ambiente de aprendizagem *Caffe*. A arquitetura *Alexnet* é uma arquitetura de rede neural convolucional proposta em Krizhevsky, Sutskever e Hinton (2012).

Zhu et al. (2016) propuseram a arquitetura apresentada na Figura 16 (b) que trabalha com imagem RBG e quatro tipos diferentes de tarefas: classificação, previsão do formato da mão, localização do ponto de referência da palma e do pulso.

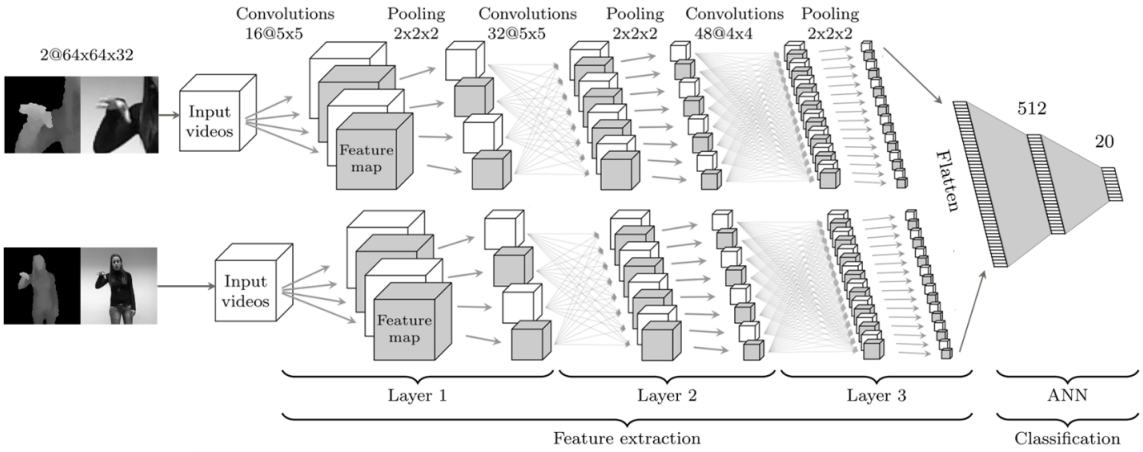
Pigou et al. (2015) propuseram a arquitetura apresentada na Figura 16 (c) que trabalha com vídeo de profundidade em escala de cinza e dois tipos diferentes de redes neurais convolucionais: uma que extrai características da mão e outra do corpo.



(a)



(b)



(c)

Figura 16 – Ilustração das arquiteturas selecionadas com base na revisão bibliográfica. (a) Arquitetura utilizada no trabalho de Kang, Tripathi e Nguyen (2015). (b) Arquitetura utilizada no trabalho de Zhu et al. (2016). (c) Arquitetura utilizada no trabalho de Pigou et al. (2015).

Como as arquiteturas selecionadas apresentavam algumas características fora do escopo deste trabalho, para utilizá-las foram realizadas as adaptações descritas nas Tabelas 3, 4 e 5. As nomenclaturas das arquiteturas adaptadas são denominadas em todo o trabalho da seguinte forma:

- Arquitetura #1: arquitetura *Alexnet*;

- Arquitetura #2: adaptação da arquitetura, proposta por Zhu et al. (2016);
- Arquitetura #3: adaptação da arquitetura, proposta por Pigou et al. (2015).

Tabela 3 – Adaptações realizadas na arquitetura utilizada em Kang, Tripathi e Nguyen (2015)

Adaptação	**Arquitetura utilizada por Kang, Tripathi e Nguyen (2015)**	**Arquitetura #1**
Entrada da rede	1@256x256x1	1@135x139x1
Camadas 1 e 2	Agrupamento seguido de normalização	Normalização seguida de agrupamento
Saída da rede	31	61

Tabela 4 – Adaptações realizadas na arquitetura utilizada em Zhu et al. (2016).

Adaptação	**Arquitetura proposta por Zhu et al. (2016)**	**Arquitetura #2**
Entrada da rede	1@40x40x3	1@135x139x1
Multi-tarefa	Sim	Não
Tarefas executadas	Classificação Previsão do formato da mão Localização do ponto de referência da palma Localização do ponto de referência do pulso	Classificação
Saída da rede	24	61

Tabela 5 – Adaptações realizadas na arquitetura utilizada em Pigou et al. (2015).

Adaptação	**Arquitetura proposta por Pigou et al. (2015)**	**Arquitetura #3**
Entrada da rede	2@64x64x32	1@135x139x1
Quantidade de redes	2	1
Finalidade de cada rede	Extrair características da mão Extrair características do corpo	Extrair características da mão
Concatenação das saídas da rede	Sim	Não
Saída	20	61

A Figura 17 (a) ilustra a composição da arquitetura #1, com maior complexidade entre as três, composta por oito camadas, das quais cinco são compostas por camadas de convolução com

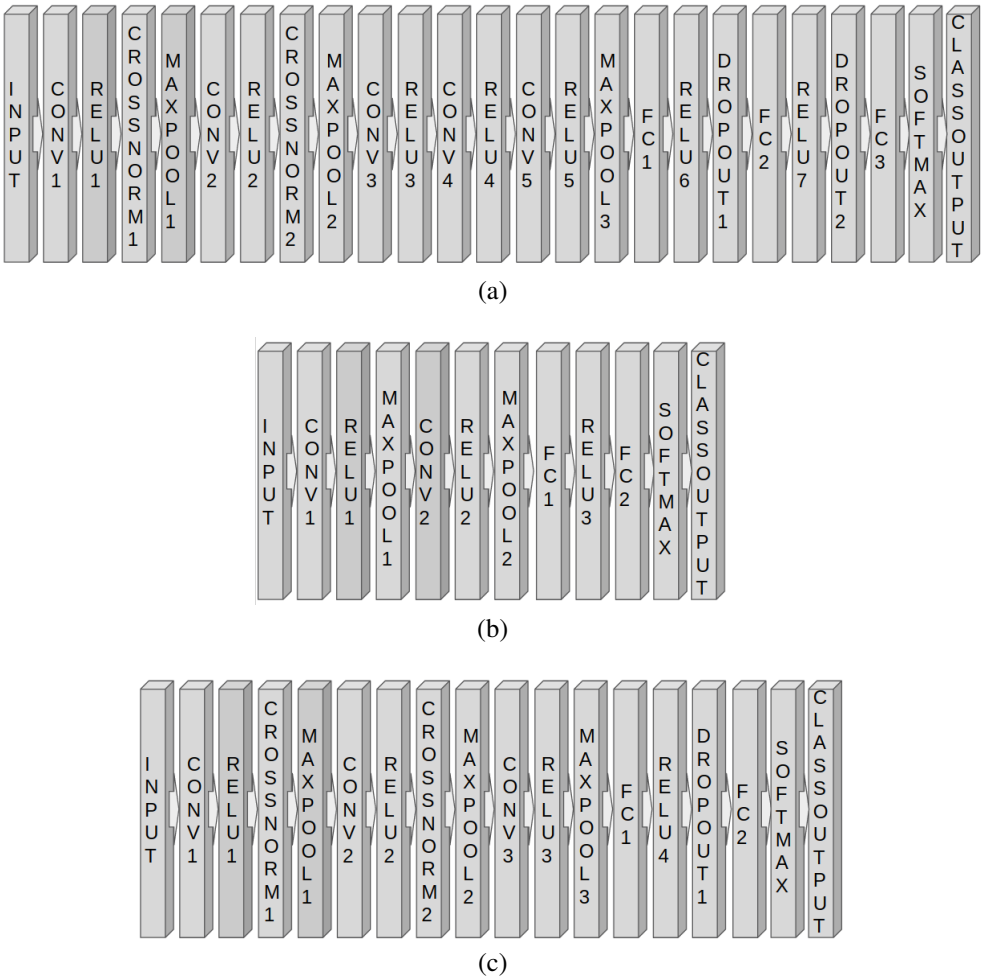


Figura 17 – Ilustração das arquiteturas utilizadas no treinamento das redes neurais convolucionais. (a) Arquitetura #1. (b) Arquitetura #2. (c) Arquitetura #3.

funções de ativação ReLU. As camadas de agrupamento de máximo estão após as camadas de convolução 1, 2 e 5. Após as camadas de agrupamento 1 e 2, estão as camadas de normalização de resposta. As três últimas camadas são compostas por camadas completamente conectadas, das quais duas possuem função de ativação ReLU e usam a técnica de regularização *dropout*. Na saída da rede, há uma camada de *softmax* e uma camada de classificação.

A Figura 17 (b) ilustra a composição da arquitetura #2, com menor complexidade entre as três, composta por quatro camadas, das quais duas são compostas por camadas de convolução com funções de ativação ReLU e camadas de agrupamento de máximo. As duas últimas camadas são compostas por camadas completamente conectadas, das quais uma possui função de ativação ReLU. Na saída da rede, há uma camada de *softmax* e uma camada de classificação.

A Figura 17 (c) ilustra a composição da arquitetura #3, com complexidade intermediária entre as três, composta por cinco camadas, das quais três são compostas por camadas de convolução com funções de ativação ReLU e camadas de agrupamento de máximo. Antes das camadas de agrupamento 1 e 2 estão as camadas de normalização de resposta. As duas últimas camadas são compostas por camadas completamente conectadas, das quais um possui função de ativação ReLU e usa a técnica de regularização *dropout*. Na saída da rede, há uma camada de *softmax* e

uma camada de classificação.

4.2.3 Treinamento das redes neurais convolucionais

Os valores para os hiperparâmetros taxa de aprendizagem, tamanho do mini-lote, momento e decaimento da taxa de aprendizagem utilizados na fase de ajuste dos hiperparâmetros estão listados na Tabela 6. A escolha dos valores mais adequados para o treinamento das redes foi realizada com base na análise das curvas de aprendizagem geradas. O critério de parada de treinamento foi estabelecido em 500 épocas.

Tabela 6 – Valores de hiperparâmetros testados.

Hiperparâmetro	**Valor**
Taxa de aprendizagem	0,1 0,01 0,001
Tamanho do mini-lote	64 128 256
Momento	0,8 0,9 1,0
Decaimento da taxa de aprendizagem	*none* *piecewise*
Número de épocas	500

Os critérios estabelecidos para escolha dos valores dos hiperparâmetros foram convergência mais rápida quando as curvas eram diferentes e tempo de processamento para realizar o treinamento quando as curvas eram muito similares. A Figura 18 ilustra o comportamento dos valores dos hiperparâmetros utilizados no ajuste.

Para as arquiteturas #1, #2 e #3, os hiperparâmetros selecionados foram: mini lote de 128, momento de 0,9 e sem decaimento da taxa de aprendizagem. Com relação a taxa de aprendizagem, os valores foram: 0,01 para as arquiteturas #1 e #3 e 0,001 para a arquitetura #2.

O treinamento das redes neurais convolucionais foi realizado com o conjunto de treinamento. Em todos os treinamentos foi utilizado o algoritmo de otimização SGDM. A Figura 19 apresenta os treinamentos realizados com as três arquiteturas de redes neurais convolucionais para obtenção de três modelos com capacidade de classificar as configurações de mão da língua brasileira de sinais.

O próximo passo adotado foi realizar os treinamentos aplicando as técnicas de regularização da seguinte forma: *dropout*, L2 e *dropout* + L2. A técnica de *dropout* foi aplicada logo após a primeira camada completamente conectada, com probabilidade de 0,5% de exclusão temporária de um neurônio e a técnica de L2 foi utilizada com o fator de 0,0001.

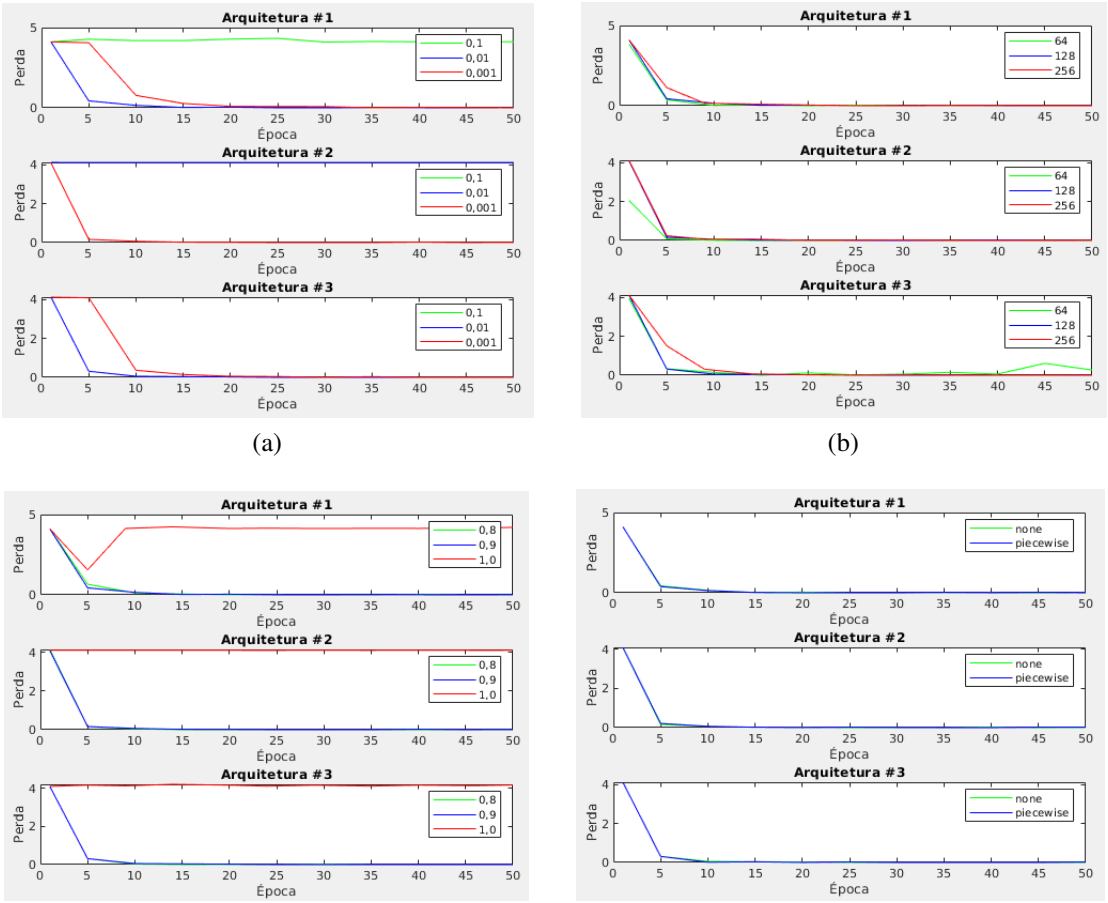
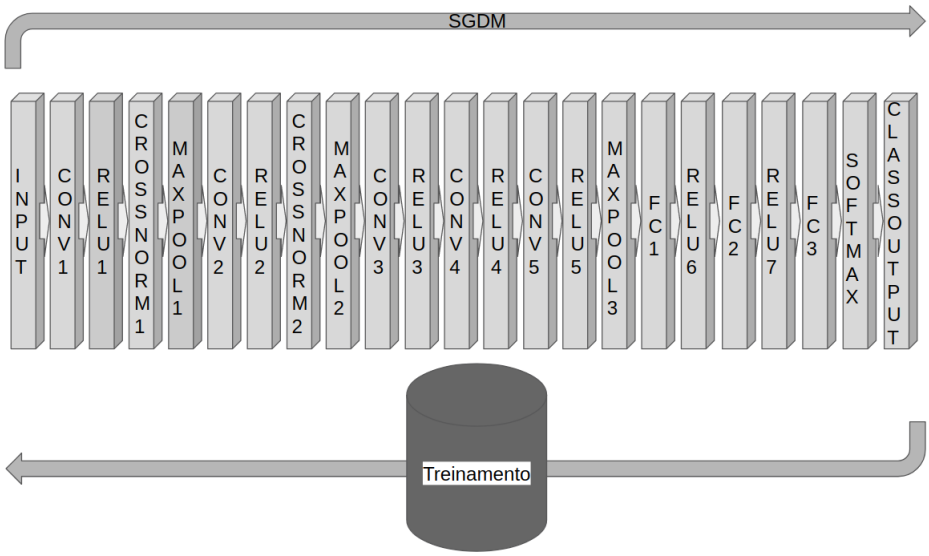


Figura 18 – Ilustração das curvas de aprendizagem em função dos hiperparâmetros. (a) Taxa de aprendizagem. (b) Tamanho do mini lote. (c) Momento. (d) Decaimento da taxa de aprendizagem.

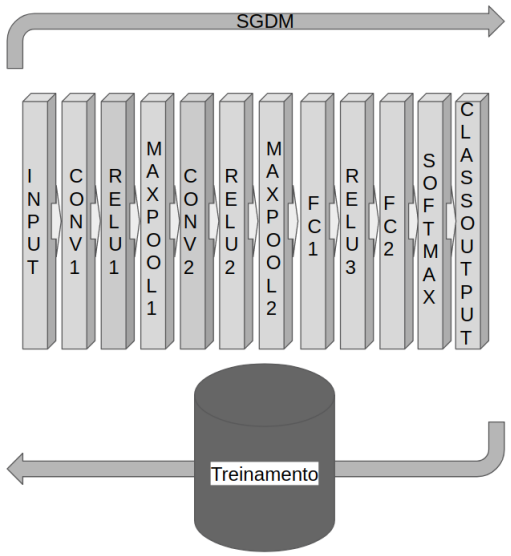
As Figuras 20, 21 e 22 apresentam os treinamentos realizados com as três arquiteturas de redes neurais convolucionais e a utilização das técnicas de regularização *dropout*, L2 e *dropout* + L2 para obtenção de nove modelos com capacidade de classificar as configurações de mão da língua brasileira de sinais.

O tempo de processamento de cada treinamento realizado com as arquiteturas e as técnicas de regularização está descrito na Tabela 7. O tempo de processamento dos treinamentos realizados somente com a arquitetura foi maior para as arquiteturas #1, #2 e #3, respectivamente. Com a utilização das técnicas, o treinamento com maior duração, para a arquitetura #1, foi aplicando *dropout* + L2 e o com menor duração foi aplicando L2. Para as arquiteturas #2 e #3, o treinamento com maior duração foi com a técnica de *dropout* e o com menor duração com as técnicas de *dropout* + L2.

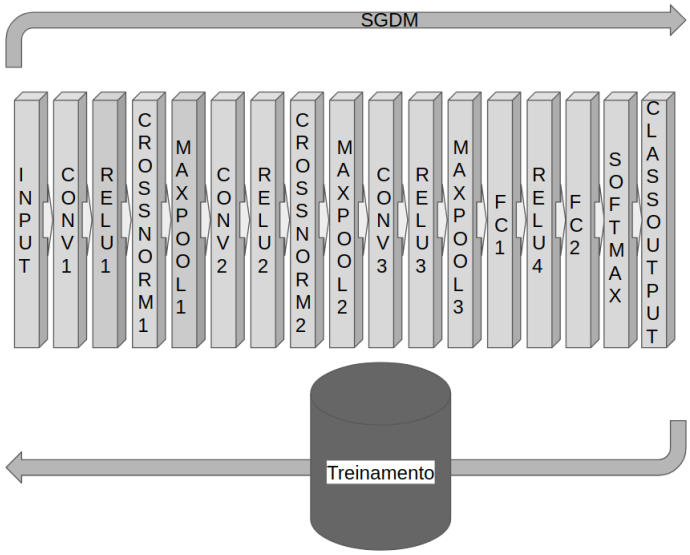
A utilização da técnica de *dropout*, em todas as arquiteturas, aumentou o tempo de processamento quando comparado aos treinamentos sem utilização das técnicas. A utilização da técnica de L2 manteve o tempo de processamento muito próximo aos treinamentos sem técnicas. A utilização das técnicas *dropout* + L2 aumentou o tempo de processamento para a arquitetura #1 e reduziu para as arquiteturas #2 e #3.



(a)

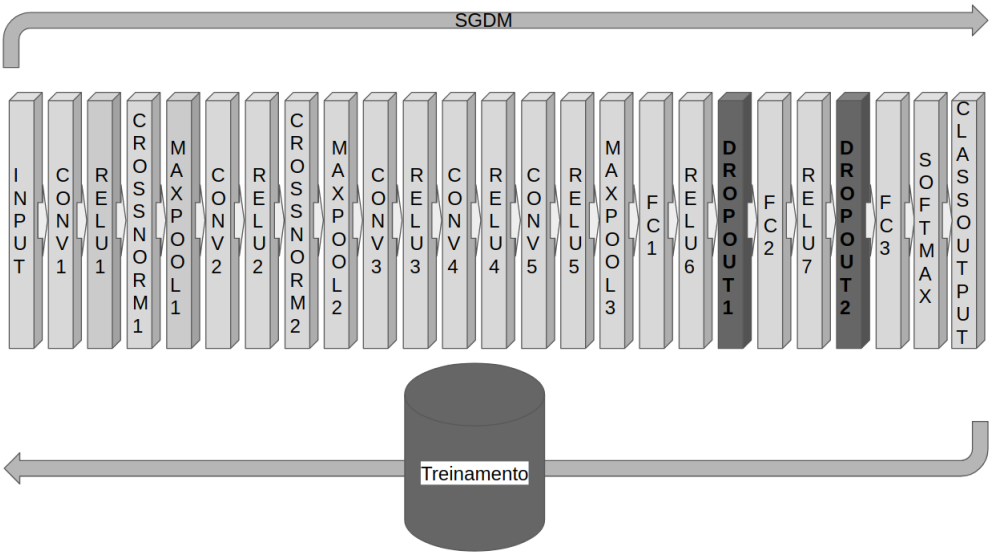


(b)

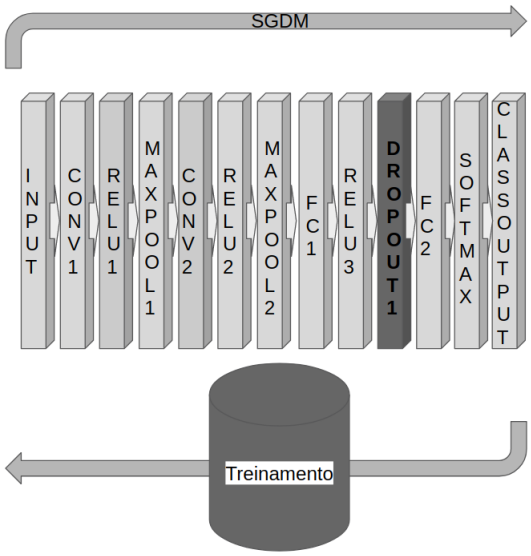


(c)

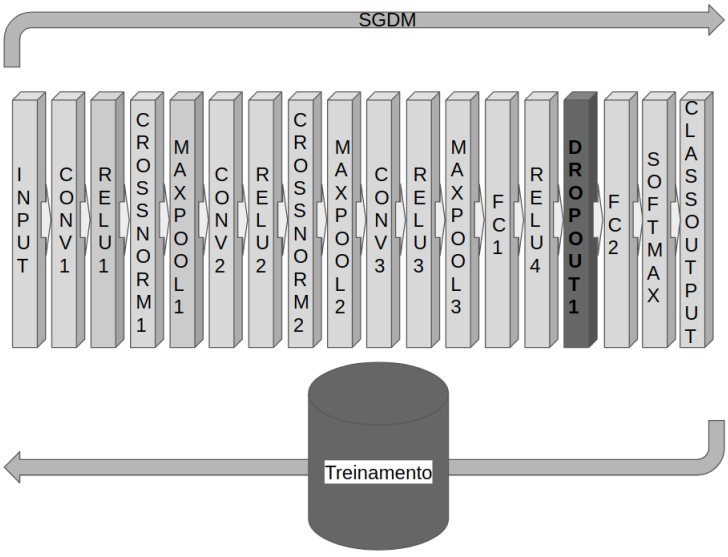
Figura 19 – Ilustração dos treinamentos realizados com as arquiteturas de redes neurais convolucionais. (a) Arquitetura #1. (b) Arquitetura #2. (c) Arquitetura #3.



(a)

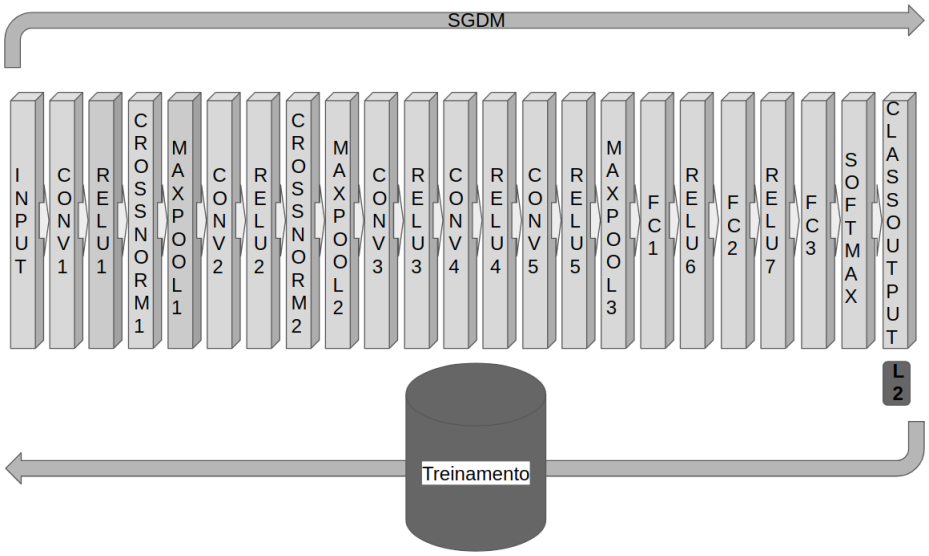


(b)

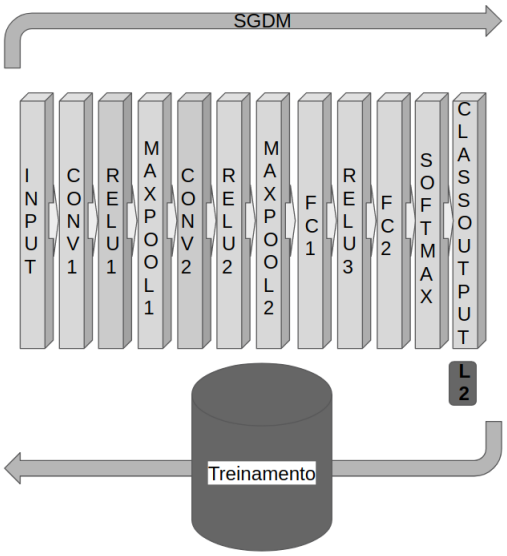


(c)

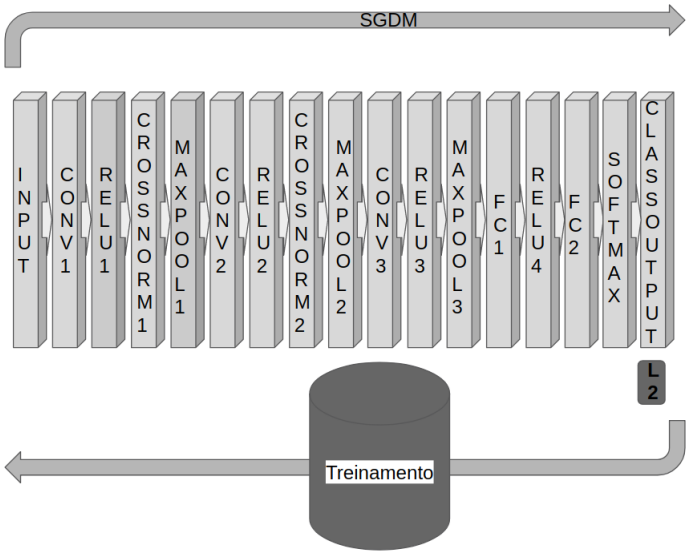
Figura 20 – Ilustração dos treinamentos realizados com as arquiteturas de redes neurais convolucionais utilizando a técnica de regularização de *dropout*. (a) Arquitetura #1. (b) Arquitetura #2. (c) Arquitetura #3.



(a)



(b)



(c)

Figura 21 – Ilustração dos treinamentos realizados com as arquiteturas de redes neurais convolucionais utilizando a técnica de regularização de L2. (a) Arquitetura #1. (b) Arquitetura #2. (c) Arquitetura #3.

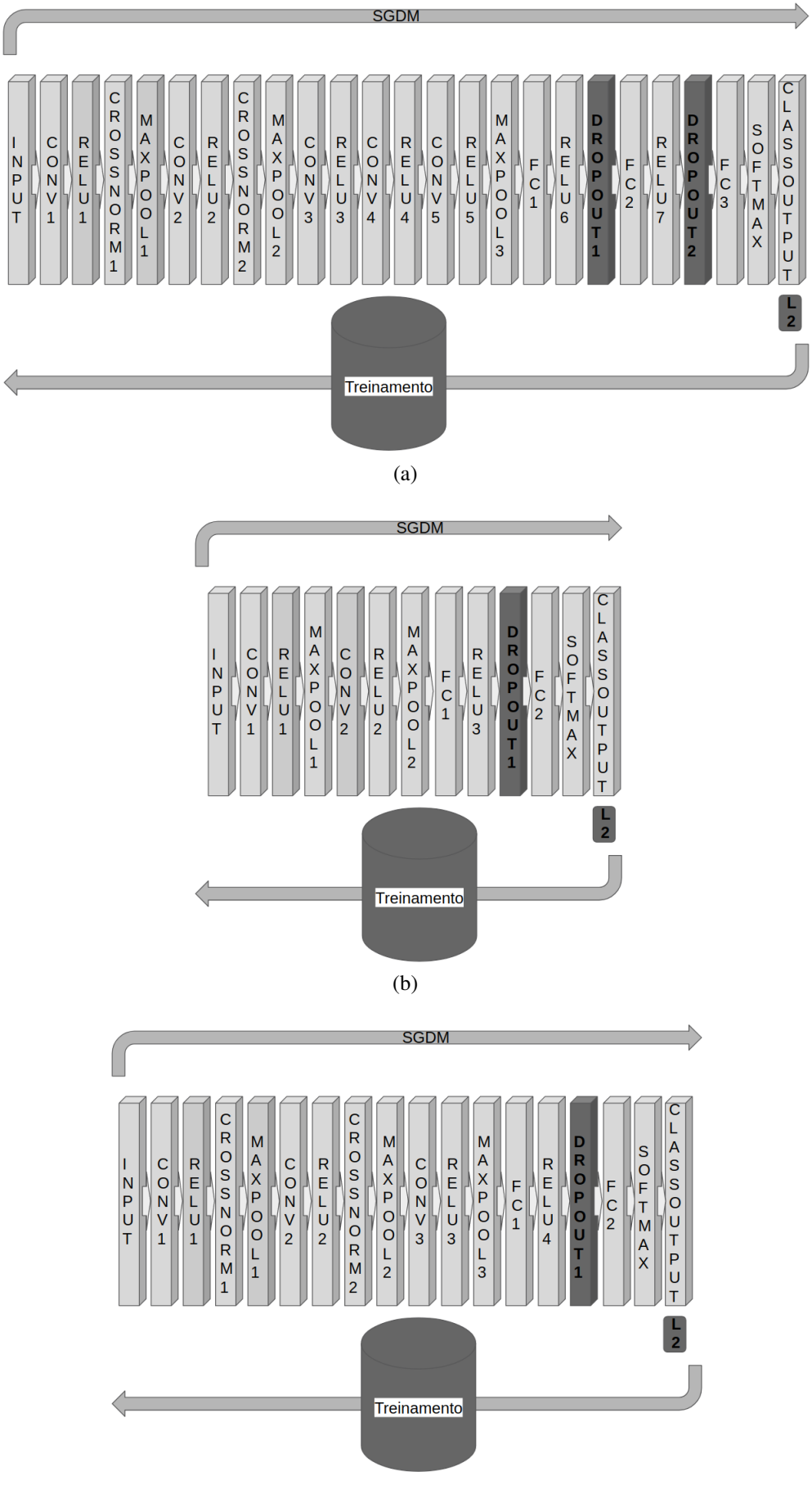


Figura 22 – Ilustração dos treinamentos realizados com as arquiteturas de redes neurais convolucionais utilizando as técnicas de regularização *dropout* + L2. (a) Arquitetura #1. (b) Arquitetura #2. (c) Arquitetura #3.

Tabela 7 – Tempo de processamento dos treinamentos realizados

Arquitetura	Técnica de regularização	Duração do treinamento
Arquitetura #1	Dropout + L2	08:21:50
	L2	07:50:13
	Dropout	07:51:58
	-	07:31:31
Arquitetura #2	Dropout + L2	05:53:17
	L2	06:17:20
	Dropout	07:13:20
	-	06:16:42
Arquitetura #3	Dropout + L2	05:32:52
	L2	05:35:51
	Dropout	06:47:40
	-	05:42:26

4.2.4 Teste dos modelos

A avaliação de modelos consiste no processo de verificação de como um modelo classifica uma imagem. A ferramenta utilizada para avaliação de modelos é a matriz de confusão ilustrada na Figura 23. Essa matriz consiste de linhas, que representam as classes reais das imagens (P e N), e colunas, que representam as classificações previstas por um modelo (P' e N') (PATTERSON; GIBSON, 2017; BEYELER, 2017; BRUCE; BRUCE, 2017).

	P' (Previsto)	N' (Previsto)
P (Real)	Verdadeiro Positivo	Falso Negativo
N (Real)	Falso Positivo	Verdadeiro Negativo

Figura 23 – Ilustração da matriz de confusão utilizada para avaliar o desempenho de modelos.

Fonte – Adaptada de Patterson e Gibson (2017).

Uma classificação é considerada verdadeira positiva (VP) quando seu valor real é positivo (P) e a classificação prevista pelo modelo é positiva (P'). Por outro lado, uma classificação é considerada falsa positiva (FP) quando seu valor real é negativo (N) e a classificação prevista pelo modelo é positiva (P').

Uma classificação é considerada falsa negativa (FN) quando seu valor real é positivo (P) e a classificação prevista pelo modelo é negativa (N'). Por outro lado, uma classificação é considerada verdadeira negativa (VN) quando seu valor real é negativo (N) e a classificação prevista pelo modelo é negativa (N'). Em testes de hipóteses estatísticas, os falsos positivos são

conhecidos como erro do tipo I e os falsos negativos como erro do tipo II. A partir da matriz de confusão, é possível calcular diferentes avaliações do modelo com base nas combinações citadas.

Para testar a capacidade de generalização dos modelos gerados na fase de treinamento foi utilizado o conjunto de teste com imagens não conhecidas pelos modelos. Para cada um dos doze modelos gerados (3 modelos sem uso de técnica de regularização mais os 9 modelos utilizando técnicas de regularização), foi calculada a acurácia. A Figura 24 apresenta o processo realizado na etapa de teste dos modelos. As nomenclaturas dos modelos são denominadas em todo o trabalho da seguinte forma:

- Modelo #1: treinado com a arquitetura #1;
- Modelo #2: treinado com a arquitetura #1 e a técnica de *dropout*;
- Modelo #3: treinado com a arquitetura #1 e a técnica de L2;
- Modelo #4: treinado com a arquitetura #1 e as técnicas de *dropout* + L2;
- Modelo #5: treinado com a arquitetura #2;
- Modelo #6: treinado com a arquitetura #2 e a técnica de *dropout*;
- Modelo #7: treinado com a arquitetura #2 e a técnica de L2;
- Modelo #8: treinado com a arquitetura #2 e as técnicas de *dropout* + L2;
- Modelo #9: treinado com a arquitetura #3;
- Modelo #10: treinado com a arquitetura #3 e a técnica de *dropout*;
- Modelo #11: treinado com a arquitetura #3 e a técnica de L2;
- Modelo #12: treinado com a arquitetura #3 e as técnicas de *dropout* + L2.

A acurácia é uma medida que soma o número de verdadeiros positivos e verdadeiros negativos $(VP+VN)$ e divide pela soma de todas as amostras do conjunto de dados $(VP+FP+FN+VN)$ conforme ilustra a Equação (6).

$$Acurácia = \frac{(VP+VN)}{(VP+FP+FN+VN)} \tag{6}$$

A acurácia é uma medida que expressa a habilidade de um modelo classificar corretamente as amostras de todo um conjunto de dados como sendo pertencentes realmente as suas classes. Por exemplo, para um modelo de classificação das configurações de mão da língua brasileira de sinais, a acurácia é a fração de amostras classificadas corretamente de todas as configurações de mão.

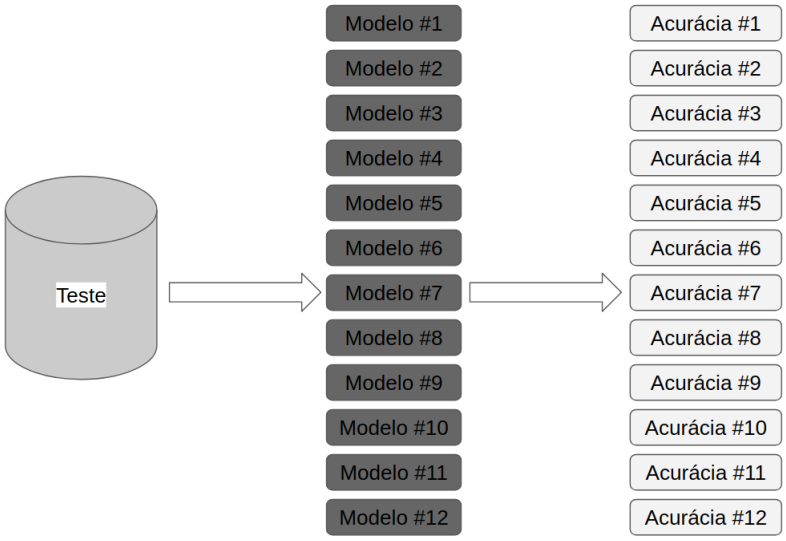


Figura 24 – Diagrama de teste dos modelos.

4.2.5 Avaliação dos modelos para seleção do melhor desempenho

Para selecionar o modelo com melhor desempenho foram utilizadas três etapas. Na primeira etapa, dos doze modelos gerados, foram selecionados os três modelos com maior valor de acurácia, cada um originalmente treinado por uma das arquiteturas utilizadas neste trabalho.

Na segunda etapa, os três modelos selecionados na etapa anterior, foram avaliados através da taxa de sensibilidade e análise da área sob a curva ROC por configuração de mão, obtidas através das matrizes de confusão disponíveis no Apêndice MATRIZ DE CONFUSÃO.

A sensibilidade, também chamado de *recall*, taxa de verdadeiros positivos ou taxa de acerto, é uma medida que expressa a habilidade de um modelo classificar, dentro de um total de amostras de uma classe, aquelas que realmente pertencem a classe (GÉRON, 2017; HACKELING et al., 2017; BONACCORSO, 2017; CAKMAK; DAS, 2018). Por exemplo, para um modelo de classificação das configurações de mão da língua brasileira de sinais, a sensibilidade da configuração 10 expressa, as amostras pertencentes a configuração 10 que foram classificadas corretamente como configuração 10.

É uma medida que conta o número de verdadeiros positivos VP e divide pela quantidade de amostras existentes para a classe em questão $(VP+FN)$, conforme ilustra a Equação (7).

$$Sensibilidade = \frac{VP}{(VP+FN)} \tag{7}$$

A curva ROC (*Receiver Operating Characteristic* ou ROC) é um gráfico que relaciona a sensibilidade com 1 - especificidade. No eixo x é plotada 1 - especificidade e no eixo y é plotada a sensibilidade. A área sob a curva ROC é chamada de AUC e representa o desempenho esperado do classificador (GÉRON, 2017; HACKELING et al., 2017; BONACCORSO, 2017; CAKMAK; DAS, 2018).

Na Figura 25 a linha tracejada representa um classificador que realiza a classificação aleatoriamente, com um AUC de 0,5. Abaixo dessa linha o desempenho de um modelo é considerado

ruim e acima dessa linha o desempenho de um modelo é considerado bom. Um modelo que apresente uma curva ROC mais próxima do canto superior esquerdo é um modelo com o melhor desempenho.

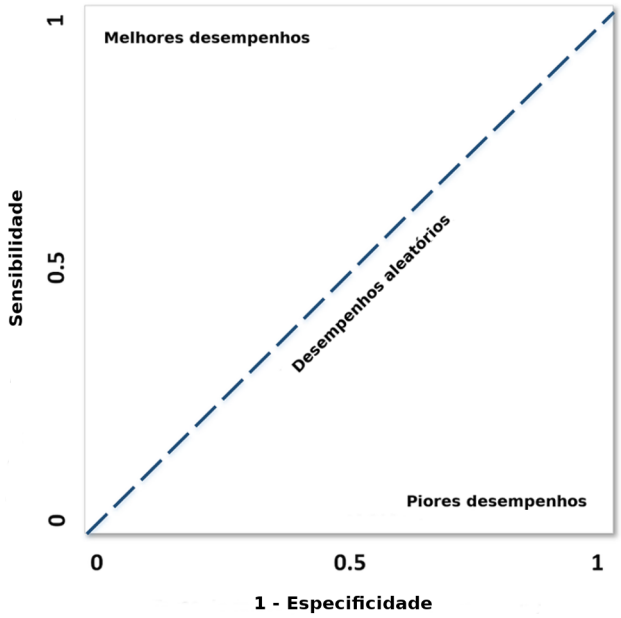


Figura 25 – Ilustração de uma curva ROC.

Fonte – Adaptada de Bonaccorso (2017).

A especificidade, também conhecida como taxa de verdadeiros negativos, é uma medida que expressa a habilidade de um modelo classificar os casos negativos corretamente. Por exemplo, para um modelo de classificação das configurações de mão da língua brasileira de sinais, a especificidade da configuração 61 expressa, as amostras não pertencentes a configuração 61 que não foram classificadas como configuração 61, ou amostras não pertencentes a configuração 61 que foram classificadas como configuração 61.

É uma medida que conta o número de verdadeiros negativos VN e divide pelo número total de negativos $(VN + FP)$ conforme ilustra na Equação (8).

$$Especifidade = \frac{VN}{(VN + FP)} \tag{8}$$

Na terceira e última etapa, os três modelos foram analisados conforme a taxa de erro de cada configuração de mão, obtidas através das matrizes de confusão presentes no Apêndice MATRIZ DE CONFUSÃO. Para cada arquitetura utilizada é feito um comparativo com relação as três maiores taxas de erro de classificação. Para alguns modelos, a taxa de erro de classificação foi a mesma para duas ou mais configurações. Neste caso, as taxas de erro de classificação são apresentadas para cada uma das configurações.

O modelo que apresentou maior acurácia média, maior sensibilidade e área sob a curva ROC por configuração de mão e menor taxa de erro também por configuração de mão, foi o modelo selecionado.

4.2.6 Análise comparativa do modelo selecionado com o estado da arte

O modelo selecionado no item anterior teve seu desempenho comparado com o desempenho apresentado pela técnica dos k-vizinhos mais próximos desenvolvida no trabalho de Costa Filho et al. (2017). As medidas utilizadas para comparação dos modelos foram: acurácia para comparar o resultado geral, sensibilidade, precisão e F1 *score* para comparar o resultado por configuração de mão. Com relação aos erros de classificação, foram consideradas as três maiores taxas de erro alcançadas pelo modelo obtido com os k-vizinhos mais próximos para comparação dos resultados.

A precisão é uma medida que expressa a habilidade de um modelo classificar, dentro de um total de amostras de uma classe somadas aquelas que foram classificadas equivocadamente como pertencentes a essa classe, aquelas que realmente pertencem a classe. Por exemplo, para um modelo de classificação das configurações de mão da língua brasileira de sinais, a precisão da configuração 1 expressa, as amostras pertencentes a configuração 1 que foram classificadas como configuração 1, em um total de amostras da configuração 1 somadas as amostras que foram classificadas como configuração 1.

É uma medida que conta o número de verdadeiros positivos VP e divide, pela quantidade de amostras existentes para a classe somadas as amostras classificadas equivocadamente como pertencentes a essa classe $(VP+FP)$, conforme ilustra a Equação (9).

$$Precisão = \frac{VP}{(VP+FP)} \tag{9}$$

O F1, também chamado de F1 *score*, F-*score* ou F-*measure*, é uma média harmônica da precisão e da sensibilidade. Essa medida multiplica duas vezes o número de verdadeiros positivos VP e divide pelo soma de duas vezes os verdadeiros positivos mais os falsos positivos e os falsos negativos $(2VP+FP+FN)$ conforme ilustra a Equação (10).

É uma medida que pontua entre 0,0 e 1,0, sendo a pior pontuação 0,0 e a melhor pontuação 1,0. Em aprendizagem de máquina é uma medida usada para pontuar o desempenho de um modelo de maneira geral e trabalha bem com conjunto de dados que possuem classes desproporcionais.

$$F1 = \frac{2VP}{(2VP+FP+FN)} \tag{10}$$

A significância estatística das diferenças de desempenho, para acurácia, dos referidos modelos foi analisada aplicando o teste qui-quadrado de Person. Para aplicar o teste, uma tabela de contingência 2×2 foi utilizada, onde cada linha da tabela corresponde ao classificador, k-vizinhos mais próximos e rede neural convolucional, e cada coluna a classificação realizada, correta e incorreta.

A hipótese nula (H_0: Não há diferença entre a técnica de rede neural convolucional e a técnica dos k-vizinhos mais próximos ao classificar as configurações de mão da língua brasileira de sinais) é rejeitada se $\chi^2 \geq \chi^2_{crítico}$. O nível de significância estabelecido foi de 99% ($\alpha = 0,01$).

Para um grau de liberdade ($GL = (L-1) \times (M-1)$) o valor crítico, de acordo com a tabela presente em Corder e Foreman (2014), é $6,63$. O valor de χ^2 é calculado pela Equação (11).

$$\chi^2 = \frac{(O_{11} - E_{11})^2}{E_{11}} + \frac{(O_{12} - E_{12})^2}{E_{12}} + ... + \frac{(O_{LM} - E_{LM})^2}{E_{LM}} \tag{11}$$

em que:

O: frequências observadas;

E: frequências esperadas;

L: número de linhas da tabela de contingência;

M: número de colunas da tabela de contingência.

5 RESULTADOS E DISCUSSÕES

Para avaliar se a utilização de redes neurais convolucionais melhora a classificação das configurações de mão da língua brasileira de sinais, foi necessário comparar os modelos gerados com essa técnica de classificação. Neste capítulo, são apresentados os desempenhos de cada modelo e as etapas de seleção do modelo com maior capacidade de generalização. Esse modelo, foi então, comparado ao resultado presente no trabalho de Costa Filho et al. (2017), treinado com o classificador k-vizinhos mais próximos.

5.1 Resultado dos modelos

O gráfico da Figura 26 apresenta o desempenho de todos os modelos avaliados na fase de teste com relação a acurácia. As barras em azul representam os resultados dos modelos utilizando apenas as arquiteturas de redes neurais convolucionais.

O modelo #1 apresentou capacidade de generalização superior aos modelos #5 e #9. Ao analisar as arquiteturas dos modelos #1 e #5, a semelhança encontrada entre elas foi com relação a camada de agrupamento, com mesmo tamanho de filtro e *stride*. Ao analisar as arquiteturas dos modelos #5 e #9, embora essas arquiteturas tenham muitas semelhanças entre si com relação às camadas da rede, a arquitetura do modelo #9 possui na camada de agrupamento um tamanho de filtro menor.

Com isso, ao se definir uma arquitetura deve ser levado em consideração a complexidade dessa arquitetura e os hiperparâmetros em nível de topologia de rede, já que, neste trabalho, houve uma melhora de 1,79% no desempenho do modelo #1, quando comparado ao modelo #5. Ao realizar essa comparação entre os modelos #1 e #9, a melhora foi de 12,11%.

As barras em vermelho representam os resultados dos modelos utilizando a técnica de *dropout*. Para os três modelos a técnica de *dropout* aumentou o desempenho. Ao comparar o desempenho do modelo #2 com o #1, houve uma melhora de 1,08%. Ao realizar a comparação entre os modelos #6 e #5, a melhora foi de 1,42%. Já entre os modelos #10 e #9, a melhora foi de 0,63%.

As barras em amarelo representam os resultados dos modelos utilizando a técnica de L2. Somente para o modelo #11 a técnica de L2 aumentou o desempenho. Ao comparar o desempenho do modelo #11 com o #9, a melhora foi de 9,61%.

As barras em roxo representam os resultados dos modelos utilizando as técnicas de *dropout* + L2. Aplicar as duas técnicas de regularização de *dropout* + L2 aumentou o desempenho dos três modelos. Ao comparar o desempenho do modelo #4 com o #1, a melhora foi de 1,09%. Ao realizar essa comparação entre os modelos #8 e #5, a melhora foi 1,60%. Já entre os modelos #12 e #9, a melhora foi de 11,37%.

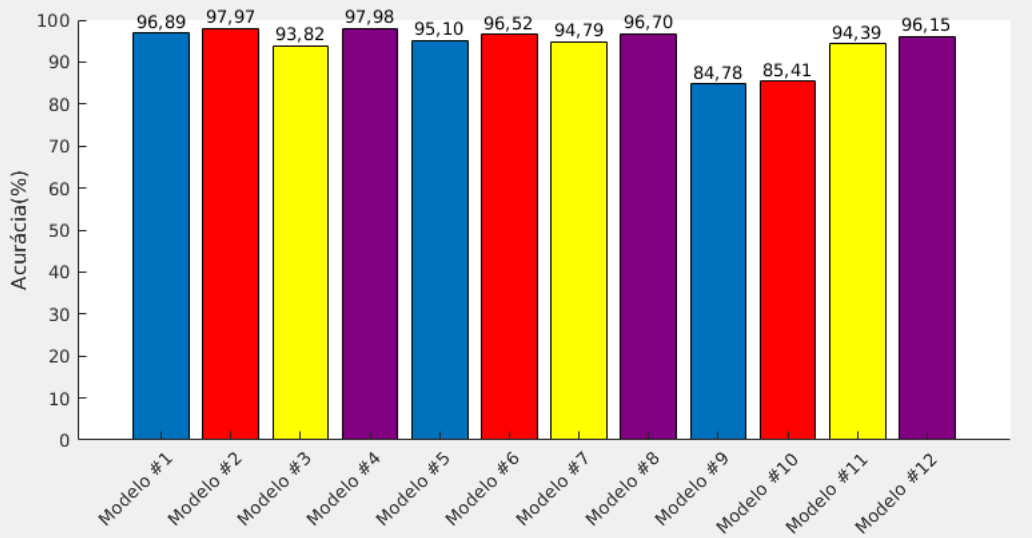


Figura 26 – Gráfico dos valores de acurácia para os doze modelos gerados com redes neurais convolucionais.

5.2 Seleção do modelo com melhor desempenho

Na primeira etapa foram selecionados os modelos #4, #8 e #12. Na segunda etapa, os modelos #4, #8 e #12 foram avaliados através da taxa de sensibilidade por configuração de mão. Para comparar as taxas de sensibilidade dos três modelos foi criada a Tabela 8.

O modelo #4 conseguiu 100% de sensibilidade para 22 configurações de mão. Isso aconteceu somente para dez configurações para o modelo #8 e para sete configurações para o modelo #12.

O modelo #4 apresentou taxas de sensibilidade maiores dos que os outros dois modelos para 28 configurações de mão. Esse comportamento também foi observado no modelo #8, para quatro configurações e no modelo #12, para quatro configurações.

Também houve semelhanças de taxa de sensibilidade ao analisar dois modelos. Para dez configurações de mão, os modelos #4 e #8 apresentaram as mesmas taxas de sensibilidade. Esse comportamento também se repetiu para os modelos #4 e #12, para cinco configurações e para os modelos #8 e #12, para duas configurações. Ainda houve o caso de taxas de sensibilidade semelhantes entre os três modelos. Isso aconteceu para oito configurações.

Esses resultados mostram que o modelo #4 apresentou taxas de sensibilidade superiores ou iguais aos demais modelos para 51 configurações de mão. Para o modelo #8 isso aconteceu para 24 configurações e para o modelo #12 somente para 19.

Tabela 8 – Resultado dos valores de sensibilidade para as 61 configurações de mão.

Configuração de mão	**Imagem**	**Sensibilidade**		
		Modelo #4	**Modelo #8**	**Modelo #12**
1		97%	91%	96%
2		98%	100%	100%

Tabela 8 (continua)

Configuração de mão	Imagem	Sensibilidade		
		Modelo #4	Modelo #8	Modelo #12
3		90%	93%	90%
4		95%	94%	92%
5		94%	95%	96%
6		93%	99%	96%
7		96%	97%	94%
8		99%	99%	99%
9		96%	95%	97%
10		99%	95%	89%
11		98%	95%	92%
12		96%	92%	94%
13		93%	90%	91%
14		100%	96%	90%
15		100%	100%	100%
16		99%	99%	95%
17		100%	97%	98%
18		100%	98%	97%
19		99%	95%	96%
20		96%	93%	96%
21		100%	99%	97%

Tabela 8 (continua)

Configuração de mão	Imagem	Sensibilidade		
		Modelo #4	Modelo #8	Modelo #12
22		100%	99%	97%
23		100%	100%	96%
24		100%	99%	100%
25		99%	99%	99%
26		91%	87%	90%
27		96%	90%	91%
28		98%	98%	98%
29		99%	99%	96%
30		97%	98%	96%
31		100%	99%	99%
32		95%	92%	95%
33		100%	95%	98%
34		100%	96%	97%
35		100%	100%	99%
36		97%	96%	93%
37		100%	100%	100%
38		100%	98%	97%
39		100%	100%	99%

Tabela 8 (continua)

Configuração de mão	Imagem	Sensibilidade		
		Modelo #4	Modelo #8	Modelo #12
40		99%	99%	98%
41		100%	100%	97%
42		96%	95%	94%
43		100%	96%	98%
44		99%	100%	100%
45		99%	99%	99%
46		97%	98%	96%
47		97%	97%	94%
48		98%	99%	98%
49		99%	99%	98%
50		99%	95%	96%
51		98%	95%	96%
52		100%	96%	99%
53		99%	99%	96%
54		99%	98%	95%
55		95%	92%	96%
56		100%	99%	98%
57		100%	100%	100%
58		97%	98%	99%

Tabela 8 (continua)

Configuração de mão	Imagem	Sensibilidade		
		Modelo #4	Modelo #8	Modelo #12
59		100%	94%	92%
60		96%	94%	91%
61		100%	100%	100%

Ainda na segunda etapa, os modelos #4, #8 e #12 foram avaliados através da análise da área sob a curva ROC por configuração de mão. Para comparar a área sob a curva ROC (AUC) dos três modelos foi criada a Tabela 9. O modelo #4 apresentou AUC maior que os outros dois modelos para 21 configurações de mão. A Figura 27 (a) apresenta a curva ROC da configuração de mão 1 para exemplificar esse resultado.

Já o modelo #8, apresentou AUC maior para três configurações apenas. A Figura 27 (b) apresenta a curva ROC da configuração de mão 26 para ilustrar um melhor desempenho desse modelo.

Somente uma única configuração apresentou AUC superior para o modelo #12 e a curva ROC dessa configuração de mão está ilustrada na Figura 27 (c).

Também houve semelhanças de AUCs ao analisar dois modelos. Para 11 configurações de mão, os modelos #4 e #8 apresentaram AUCs iguais. Esse comportamento também se repetiu para os modelos #4 e #12, para seis configurações e para os modelos #8 e #12, para quatro configurações. Ainda existem casos em que as AUCs das configurações são iguais para todos os modelos. Isso aconteceu para 15 configurações.

Esses resultados mostram que o modelo #4 apresentou AUCs superiores ou iguais aos demais modelos para 53 configurações de mão. Para o modelo #8, isso aconteceu para 33 configurações e para o modelo #12, somente para 26.

Tabela 9 – Resultado dos valores de área sob a curva (AUC) para as 61 configurações de mão.

Configuração de mão	Imagem	AUC		
		Modelo #4	Modelo #8	Modelo #12
1		0,9999	0,9996	0,9996
2		1	1	1
3		0,9991	0,9992	0,9979
4		0,9996	0,9996	0,9988

Tabela 9 (continua)

Configuração de mão	Imagem	AUC		
		Modelo #4	Modelo #8	Modelo #12
5		0,9999	0,9996	0,9988
6		0,9999	1	1
7		0,9999	0,9996	0,9997
8		1	1	1
9		0,9995	0,9992	0,9992
10		0,9999	0,9997	0,9994
11		1	0,9999	0,9998
12		0,9998	0,9995	0,9998
13		0,9987	0,9987	0,9993
14		1	0,9998	0,9997
15		1	1	1
16		1	0,9999	0,9999
17		1	0,9999	1
18		1	0,9999	1
19		1	0,9998	1
20		0,9991	0,9994	0,9998
21		1	1	1
22		0,9999	0,9999	0,9999

Tabela 9 (continua)

Configuração de mão	Imagem	AUC		
		Modelo #4	Modelo #8	Modelo #12
23		1	0,9999	0,9998
24		1	1	1
25		1	1	1
26		0,9980	0,9994	0,9961
27		0,9998	0,9997	0,9994
28		0,9999	0,9985	0,9989
29		1	1	0,9999
30		0,9999	0,9997	0,9997
31		1	1	1
32		0,9999	0,9997	0,9999
33		1	0,9999	1
34		1	0,9998	0,9995
35		1	1	1
36		0,9998	1	0,9999
37		1	1	1
38		1	1	0,9999
39		1	1	1
40		1	1	1
41		1	1	0,9999

Tabela 9 (continua)

Configuração de mão	Imagem	AUC		
		Modelo #4	Modelo #8	Modelo #12
42		1	0,9998	0,9996
43		1	0,9998	0,9999
44		1	1	1
45		1	1	1
46		0,9998	0,9999	0,9999
47		1	1	0,9999
48		0,9998	0,9999	0,9999
49		1	0,9999	0,9999
50		1	0,9999	0,999
51		1	1	0,9999
52		1	0,9991	0,9998
53		0,9999	1	0,9999
54		0,9999	0,9999	0,9998
55		0,9993	0,9999	0,9999
56		1	1	0,9999
57		1	1	0,9999
58		1	1	0,9999
59		1	0,9998	0,9992

Tabela 9 (continua)

Configuração de mão	Imagem	AUC		
		Modelo #4	Modelo #8	Modelo #12
60		1	0,9998	0,9992
61		1	1	1

Na terceira e última etapa, os modelos #4, #8 e #12 foram analisados conforme a taxa de erro por configuração de mão. O modelo #4 apresentou problemas para classificar 39 configurações de mão. Isso aconteceu para 51 configurações para o modelo #8 e para 54 configurações para o modelo #12.

A maior taxa de erro para o modelo #4 foi para a configuração de mão 3, com 10% de erro de classificação. A segunda maior taxa de erro foi para a configuração de mão 26, com 9% de erro de classificação. A terceira maior taxa de erro foi para as configurações de mão 6 e 13, com 7% de erro de classificação.

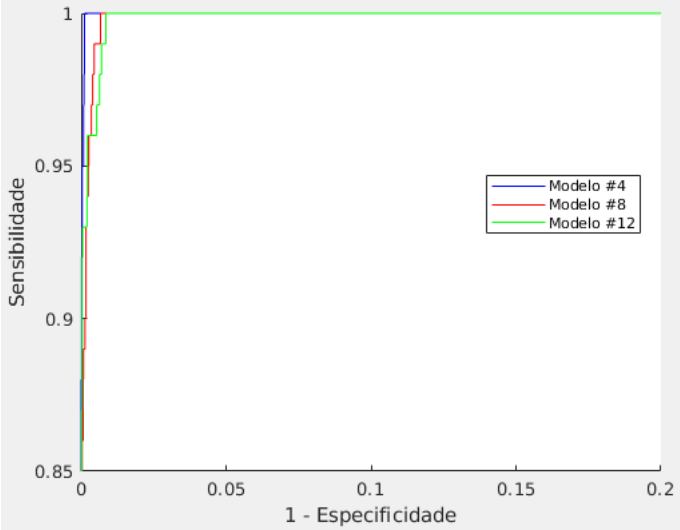
Para o modelo #8, a maior taxa de erro foi para a configuração de mão 26, com 13% de erro de classificação. As segundas maiores taxas de erros foram para as configurações de mão 13 e 27, ambas com 10% de erro de classificação. A terceira maior taxa de erro foi para a configuração de mão 1, com 9% de erro de classificação.

Para o modelo #12, a maior taxa de erro foi para a configuração de mão 10, com 11% de erro de classificação. A segunda maior taxa de erro foi para as configurações de mão 3, 14 e 26, ambas com 10% de erro de classificação. A terceira maior taxa de erro foi para as configurações de mão 13, 27 e 60, com 9% de erro de classificação.

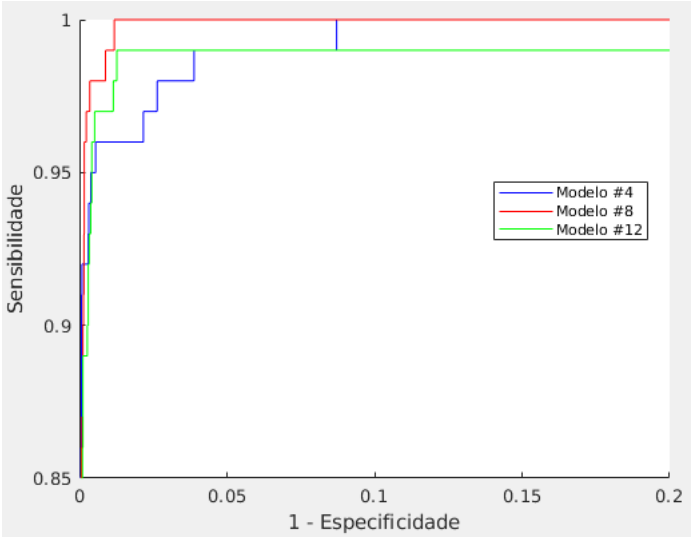
Um comportamento similar dos modelos é que os mesmos, ao receber uma determinada configuração de mão em sua entrada, preveem configurações algumas vezes similares entre si, outras vezes bem divergentes. Por exemplo, o modelo #4, reconhece a configuração de mão 1 como sendo as configurações de mão 2, 9 ou 10. Porém, o modelo #8, reconhece a configuração de mão 1 como sendo as configurações 2, 9, 10, 11, 17 ou 61. Já o modelo #12, reconhece a configuração 1 como sendo a 2, 9 ou 11 (vide Tabela 10).

Em alguns casos, o modelo #4, embora apresente a maior taxa média de acurácia, apresenta erros de classificação que não são observados nos demais modelos. Por exemplo, quando a configuração de mão 2 é apresentada ao modelo #4, o mesmo classifica a configuração 2 como sendo a configuração 1 ou 18, e esse erro de classificação não ocorre nos outros modelos (vide Tabela 10).

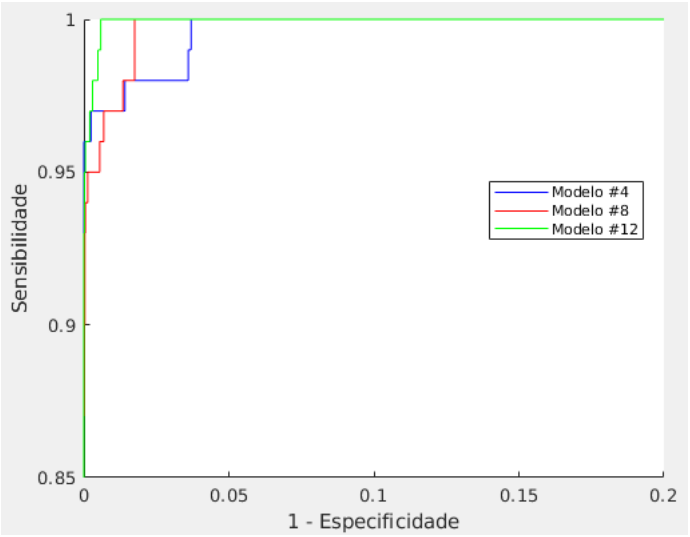
Existem configurações de mão que foram classificadas 100% de forma correta por todos os modelos. Nessa categoria, estão as configurações 15, 37, 57 e 61. Esse comportamento se repete quando dois modelos são analisados. Para os modelos #4 e #8, esse comportamento ocorre para as configurações de mão 35, 39 e 41 e para os modelos #8 e #12, para as configurações 2 e 44,conforme pode ser visto na Tabela 10.



(a)



(b)



(c)

Figura 27 – Gráfico da curva ROC que ilustra o desempenho dos modelos #4, #8 e #12, respectivamente. (a) Configuração de mão 1. (b) Configuração de mão 26. (c) Configuração de mão 20.

Por fim, existem configurações de mão que apresentam erros de classificação para todos os modelos, porém um deles apresenta uma taxa de erro de classificação menor que os demais, e muitas vezes, esse modelo não apresentou maior taxa de acurácia média. Por exemplo, para a configuração de mão 3, o modelo #4 apresentou erro de classificação de 10%, o modelo #8 apresentou erro de 7% e o modelo #12 apresentou erro de 10% (vide Tabela 10).

O melhor desempenho alcançado, para as medidas de avaliação: acurácia média do modelo, sensibilidade, área sob a curva ROC (AUC) e taxa de erro por configuração de mão, foi utilizando o modelo #4.

Tabela 10 – Resultado dos valores de erro de classificação para algumas configurações de mão.

Configuração de mão real	Imagem real	Configuração de mão prevista	Imagem prevista	Erro de classificação		
				Modelo #4	Modelo #8	Modelo #12
1		2		1%	1%	1%
		9		1%	4%	2%
		10		1%	1%	
		11			1%	1%
		17			1%	
		61			1%	
2		1		1%		
		18		1%		
3		4		10%	6%	7%
		6				1%
		9			1%	2%
35		37				1%
39		5				1%
41		40				3%

Tabela 10 (continua)

Configuração de mão real	Imagem real	Configuração de mão prevista	Imagem prevista	Erro de classificação		
				Modelo #4	Modelo #8	Modelo #12
44		47		1%		

5.3 Melhorias alcançadas com a utilização da técnica de rede neural convolucional

Com relação a acurácia, a utilização de rede neural convolucional aumentou o valor para 97,98%, valor superior ao alcançado com o classificador k-vizinhos mais próximos, de 96,31%, apresentado em Costa Filho et al. (2017).

Com relação a sensibilidade, a utilização de rede neural convolucional aumentou as taxas para 41 configurações de mão, para quatro não houve melhoras e para dezesseis as taxas se mantiveram superiores para o classificador k-vizinhos mais próximos. Na Figura 28, são apresentadas as sensibilidades por configuração de mão dos dois modelos.

Com relação a precisão, a utilização de rede neural convolucional aumentou as taxas para 37 configurações de mão, para sete não houve melhoras e para 17 as taxas se mantiveram superiores para o modelo treinado com o classificador k-vizinhos mais próximos. Na Figura 29, são apresentadas as precisões por configuração de mão dos dois modelos.

Com relação ao F1 *score*, a utilização de rede neural convolucional aumentou as taxas para 41 configurações de mão, para uma não houve melhoras e para 19 as taxas mantiveram superiores para o modelo treinado com o classificador k-vizinhos mais próximos. Na Figura 30, são apresentados os F1 *score* por configuração de mão dos dois modelos.

Com relação aos erros de classificação, as maiores taxas de erros para o modelo treinado com os k-vizinhos mais próximos foram para as configurações de mão 23, 27 e 60 com 13%, 11% e 11% de erro de classificação. Para a configuração de mão 23, o modelo treinado com rede neural convolucional não apresentou erros de classificação, para a configuração 27 o erro diminuiu para 4% e para a configuração 60 diminuiu para 4%.

Com a utilização de rede neural convolucional a etapa de teste de novas imagens não depende das imagens utilizadas na etapa de treinamento, como acontece com o classificador k-vizinhos mais próximos, que atribui uma classe para uma nova imagem a partir do ponto mais próximo entre essa nova imagem e o restante do conjunto de treinamento. Além disso, não há necessidade de uma etapa de pré-processamento de imagens para extração de características, etapa esta, necessária quando se trabalha com técnicas de aprendizagem tradicional.

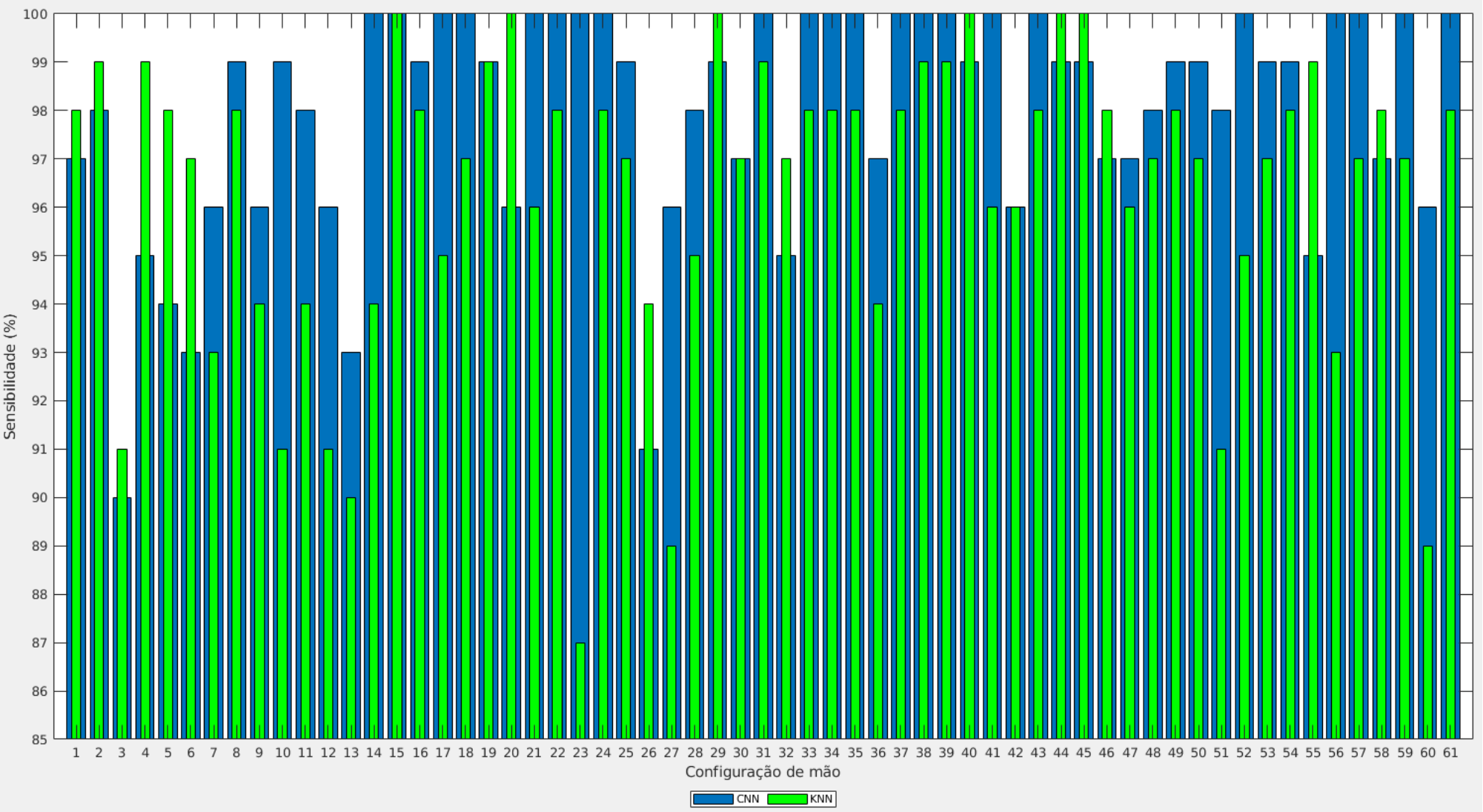


Figura 28 – Gráfico dos valores de sensibilidade para as 61 configurações de mão referente ao modelo treinado com rede neural convolucional e ao modelo treinado com o classificador k-vizinhos mais próximos apresentado em Costa Filho et al. (2017).

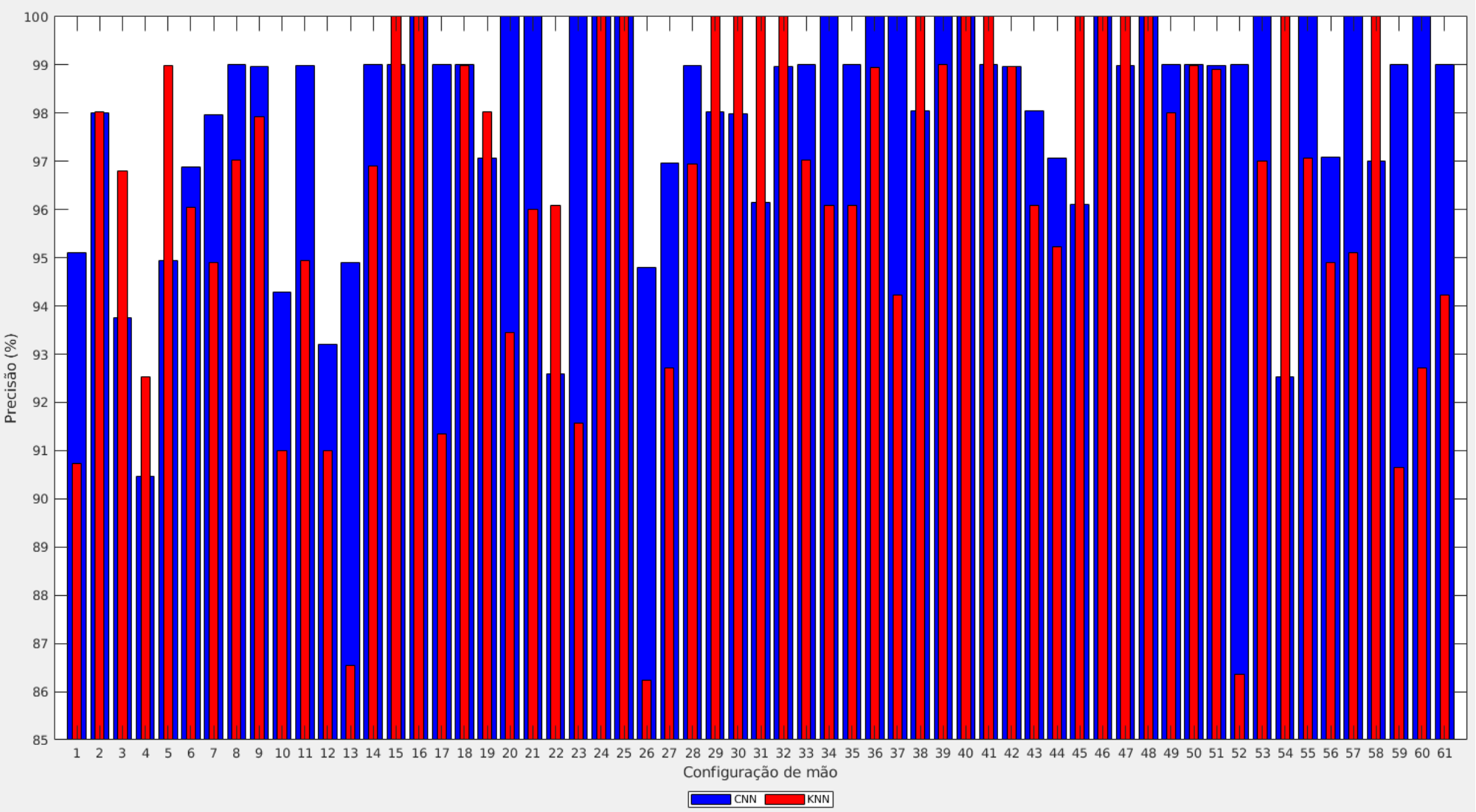


Figura 29 – Gráfico dos valores de precisão para as 61 configurações de mão referente ao modelo treinado com rede neural convolucional e ao modelo treinado com o classificador k-vizinhos mais próximos apresentado em Costa Filho et al. (2017).

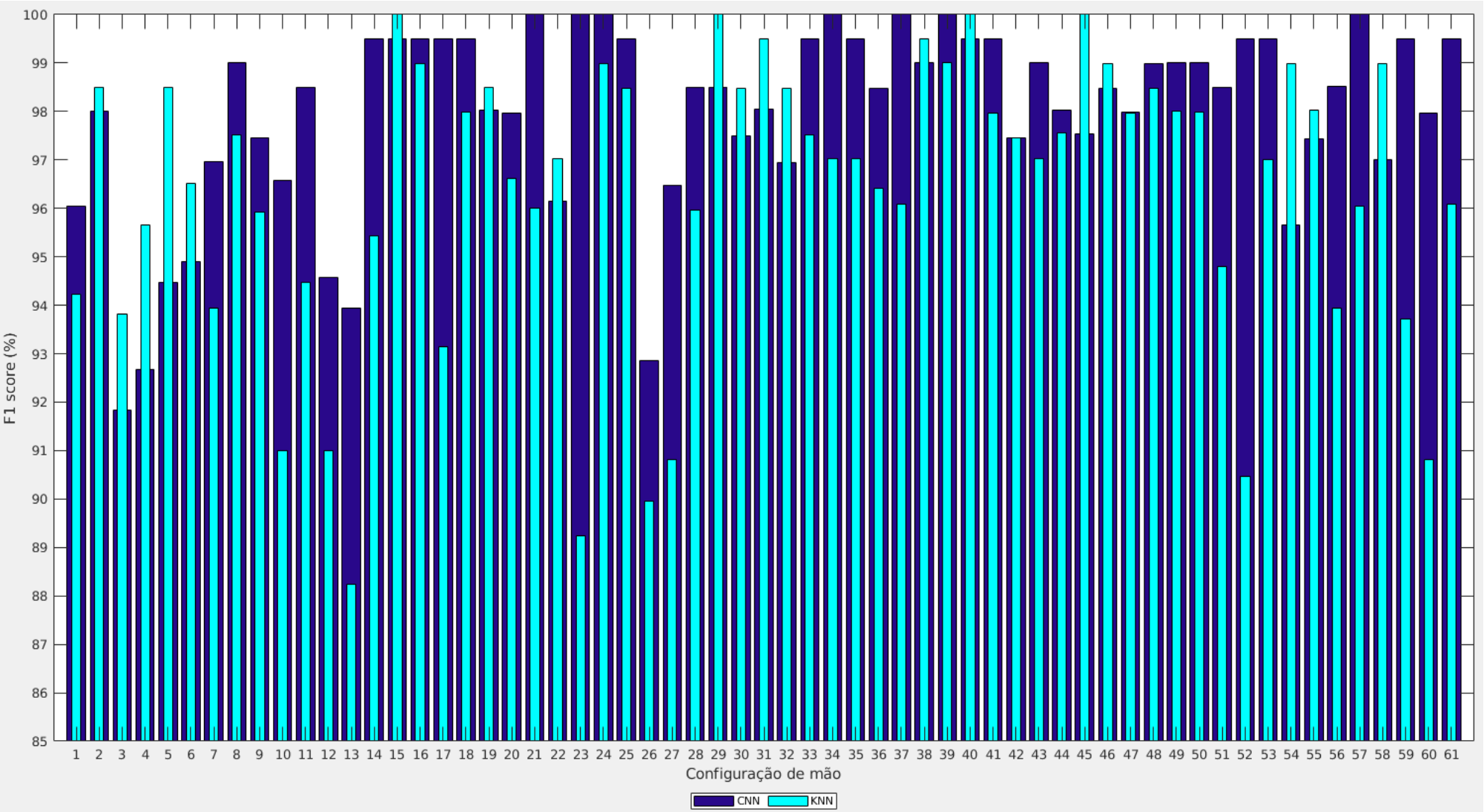


Figura 30 – Gráfico dos valores de F1 *score* para as 61 configurações de mão referente ao modelo treinado com rede neural convolucional e ao modelo treinado com o classificador k-vizinhos mais próximos apresentado em Costa Filho et al. (2017).

5.4 Resultado da análise de significância estatística

Para verificar se o aumento, em termos de acurácia, alcançado com o modelo treinado usando as redes neurais convolucionais, foi estatisticamente significativo com relação ao modelo treinado usando o k-vizinhos mais próximos, apresentado em Costa Filho et al. (2017), foi aplicado o teste qui-quadrado de Pearson.

As Tabelas 11 e 12 apresentam as tabelas de contingências com as frequências observadas e esperadas após a etapa de teste dos modelos.

Tabela 11 – Tabela de contingência com as frequências observadas.

	Correta	**Incorreta**	**Total**
K-vizinhos mais próximos	5.875	225	6.100
Rede neural convolucional	5.977	123	6.100
Total	11.852	348	12.200

Tabela 12 – Tabela de contingência com as frequências esperadas.

	Correta	**Incorreta**	**Total**
K-vizinhos mais próximos	5.926	174	6.100
Rede neural convolucional	5.926	174	6.100
Total	11.852	348	12.200

O valor de χ^2 de Pearson obtido foi $\chi^2 = 30,774$, como mostra a Equação (12).

$$\chi^2 = \frac{(5.875-5.926)^2}{5.926} + \frac{(225-174)^2}{174} + \frac{(5.977-5.926)^2}{5.926} + \frac{(123-174)^2}{174} = 30,774 \quad (12)$$

Como $\chi^2 > \chi^2_{crítico}$, a diferença entre os dois métodos, ao classificar as configurações de mão da língua brasileira de sinais, é estatisticamente significativa.

6 CONCLUSÕES

O questionamento inicial que norteou toda esta pesquisa, tinha como indagação descobrir se, a utilização de redes neurais convolucionais era uma técnica de classificação, mais promissora na geração de modelos com capacidade de classificar as configurações de mão da língua brasileira de sinais, do que a técnica tradicional, apresentada no trabalho de Costa Filho et al. (2017), já então empregada nesse tipo de tarefa de classificação com o classificador k-vizinhos mais próximos.

Para tentar identificar esse comportamento, foram traçadas estratégias de treinamento de modelos com o intuito de chegar a resultados próximos ou iguais ao já conseguido pelo classificador k-vizinhos mais próximos. A escolha de três arquiteturas com diferentes níveis de complexidade combinadas as duas técnicas de regularização foi a estratégia escolhida para tentar responder ao questionamento inicial.

Ao analisar os modelos treinados somente com as arquiteturas de redes neurais convolucionais, o modelo #1 apresentou a maior acurácia, em segundo lugar ficou o modelo #5 e em terceiro, o modelo #9. Ao analisar as arquiteturas desses modelos foram encontradas semelhanças e diferenças com relação a camada de agrupamento.

Com isso, a complexidade da arquitetura e os hiperparâmetros em nível de topologia de rede devem ser considerados no momento em que as arquiteturas estiverem sendo definidas, pois neste trabalho, houve uma melhora de 1,79% no desempenho do modelo #1 quando comparado ao modelo #5. Com relação aos modelos #1 e #9, essa melhora foi de 12,11%.

A fim de verificar o efeito das técnicas de regularização, sobre a classificação conseguida com redes neurais convolucionais, foram avaliadas as técnicas de regularização de *dropout* e L2, sozinhas e combinadas em cada uma das arquiteturas utilizadas neste trabalho.

A técnica de *dropout* aumentou o desempenho dos modelos #2, #6 e #10 em 1,08%, 1,42% e 0,63%, respectivamente. Já a utilização da técnica de L2 melhorou o desempenho referente apenas ao modelo #11, em 9,61%. A combinação das técnicas de regularização *dropout* + L2 melhorou o desempenho dos modelos #4, #8 e #12 em 1,09%, 1,60% e 11,37%.

Os três modelos com maior desempenho foram os #4, #8 e #12. Ao comparar esses três modelos com relação à sensibilidade, área sob a curva ROC e taxa de erro por configuração de mão, o modelo com maior capacidade de classificar as configurações de mão da língua brasileira de sinais, foi o #4.

Ao comparar o modelo com melhor desempenho utilizando rede neural convolucional com o modelo utilizando o classificador k-vizinhos mais próximos, das 61 configurações existentes no conjunto de dados, as taxas de sensibilidade melhoraram para 41 configurações, 4 apresentaram o mesmo valor e 16 ainda continuaram com taxas superiores para o classificador k-vizinhos mais próximos.

Com relação a precisão, as taxas melhoraram para 37 configurações de mão, 7 apresentaram

o mesmo valor e 17 foram melhores para o classificador k-vizinhos mais próximos. Com relação ao F1 *score*, as taxas melhoraram para 41 configurações, 1 apresentou o mesmo valor e 19 foram melhores para o classificador k-vizinhos mais próximos.

Com relação a acurácia, o valor aumentou para 97,98%, valor superior ao alcançado com o classificador k-vizinhos mais próximos, de 96,31%. Portanto, a aplicação de redes neurais convolucionais melhorou em 1,67% a classificação das configurações de mão da língua brasileira de sinais, sendo essa diferença estatisticamente significativa.

Os resultados são conclusivos, e afirmam a indagação realizada no início da pesquisa, de que a utilização de redes neurais convolucionais é, sim, uma técnica de classificação que gera modelos com melhor capacidade de classificar as configurações de mão da língua brasileira de sinais.

6.1 Recomendações para trabalhos futuros

A partir dos resultados alcançados, com a utilização de redes neurais convolucionais na classificação das configurações de mão da língua brasileira de sinais, as seguintes recomendações para trabalhos futuros foram diagnosticadas:

- Utilização de técnicas de agrupamento que permitam combinar as melhores taxas obtidas com redes neurais convolucionais;
- Utilização de algoritmos de otimização com aprendizagem adaptativa para comparação com os resultados utilizando SGDM;
- Utilização de técnicas de ajustes de hiperparâmetros automáticas para comparação com os resultados utilizando técnica manual;
- Utilização de técnicas de partição do conjunto de dados para comparação com os resultados utilizando a técnica *interleave*;
- Criação de uma arquitetura própria, com camadas e filtros estabelecidos experimentalmente, para comparação com os resultados obtidos com as arquiteturas utilizadas neste trabalho;
- Construção de um conjunto de dados dos sinais de Libras armazenados no formato de vídeo e aplicação de redes neurais convolucionais para classificar esse tipo de material. Alguns trabalhos levantados na revisão bibliográfica já estão trabalhando com esse tipo de classificação.

REFERÊNCIAS

ALMEIDA, S. G. M.; GUIMARAES, F. G.; RAMIREZ, J. A. Feature extraction in brazilian sign language recognition based on phonological structure and using rgb-d sensors. *Expert Systems with Applications*, v. 41, n. 16, p. 7259–7271, 2014. ISSN 0957-4174. Disponível em: <<http>://WOS:000340689700025>.

AMEEN, S.; VADERA, S. A convolutional neural network to classify american sign language fingerspelling from depth and colour images. *Expert Systems*, v. 34, n. 3, 2017. ISSN 02664720. Disponível em: <http://dx.doi.org/10.1111/exsy.12197>.

ARAUJO, L. et al. *Hands-On Convolutional Neural Networks with TensorFlow*. Packt, 2018. ISBN 9781789130331. Disponível em: <https://www.safaribooksonline.com>.

BEYELER, M. *Machine Learning for OpenCV*. Packt, 2017. ISBN 9781783980284. Disponível em: <https://www.safaribooksonline.com>.

BONACCORSO, G. *Machine Learning Algorithms*. Packt, 2017. ISBN 9781785889622. Disponível em: <https://www.safaribooksonline.com>.

BRUCE, A.; BRUCE, P. *Practical Statistics for Data Scientists*. O'Reilly Media, 2017. ISBN 9781491952962. Disponível em: <https://www.safaribooksonline.com>.

BUDUMA, N.; LOCASCIO, N. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. O'Reilly Media, 2017. ISBN 9781491925560. Disponível em: <https://www.safaribooksonline.com>.

CAKMAK, U.; DAS, S. *Hands-On Automated Machine Learning*. Packt, 2018. ISBN 9781788629898. Disponível em: <https://www.safaribooksonline.com>.

CIRESAN, D.; MEIER, U.; SCHMIDHUBER, J. Multi-column deep neural networks for image classification. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. [S.l.], 2012. p. 3642–3649.

CORDER, G.; FOREMAN, D. *Nonparametric Statistics: A Step-by-Step Approach*. Wiley, 2014. ISBN 9781118840313. Disponível em: <https://www.safaribooksonline.com>.

COSTA FILHO, C. F. F. et al. A new method for recognizing hand configurations of brazilian gesture language. In: *38th Annual International Conference of the IEEE-Engineering-in-Medicine-and-Biology-Society (EMBC)*. [s.n.], 2016. (IEEE Engineering in Medicine and Biology Society Conference Proceedings), p. 3829–3834. Disponível em: <<http>://WOS:000399823504046>.

COSTA FILHO, C. F. F. et al. A fully automatic method for recognizing hand configurations of brazilian sign language. *Revista Brasileira de Engenharia Biomedica*, v. 33, n. 1, p. 78–89, 2017. ISSN 15173151. Disponível em: <http://dx.doi.org/10.1590/2446-4740.03816>.

DANGETI, P. *Statistics for Machine Learning*. Packt Publishing Limited, 2017. ISBN 9781788295758. Disponível em: <https://www.safaribooksonline.com>.

ESCALERA, S. et al. Chalearn looking at people challange 2014: Dataset and results. In: *ECCV workshop, 2014*. [S.l.: s.n.], 2014.

ESCOBEDO, E.; CAMARA, G. A robust gesture recognition using hand local data and skeleton trajectory. In: *2015 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2015. p. 1240–1244.

ESCOBEDO, E.; CAMARA, G. A new approach for dynamic gesture recognition using skeleton trajectory representation and histograms of cumulative magnitudes. In: *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2016. p. 209–216.

GÉRON, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2017. ISBN 9781491962244. Disponível em: <https://www.safaribooksonline.com>.

GÉRON, A. *Neural network and deep learning*. O'Reilly Media, 2018. ISBN 9781492037347. Disponível em: <https://www.safaribooksonline.com>.

GODOY, V. et al. An hmm-based gesture recognition method trained on few samples. In: *2014 IEEE 26th International Conference on Tools with Artificial Intelligence*. [S.l.: s.n.], 2014. p. 640–646. ISBN 1082-3409.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. MIT Press, 2016. Disponível em: <http://www.deeplearningbook.org>.

HACKELING, G. et al. *Scikit-learn: Machine Learning Simplified*. Packt, 2017. ISBN 9781788833479. Disponível em: <https://www.safaribooksonline.com>.

HOPE, T.; RESHEFF, Y.; LIEDER, I. *Learning TensorFlow: A Guide to Building Deep Learning Systems*. O'Reilly Media, 2017. ISBN 9781491978481. Disponível em: <https://www.safaribooksonline.com>.

HUANG, J. et al. Sign language recognition using 3d convolutional neural networks. In: ____. *2015 Ieee International Conference on Multimedia and Expo*. [s.n.], 2015. (IEEE International Conference on Multimedia and Expo). ISBN 978-1-4799-7082-7. Disponível em: <<http>://WOS:000380486500051>.

KANG, B.; TRIPATHI, S.; NGUYEN, T. Q. Real-time sign language fingerspelling recognition using convolutional neural networks from depth map. In: *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. [S.l.: s.n.], 2015. p. 136–140.

KIM, P. *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence*. Apress, 2017. ISBN 9781484228456. Disponível em: <https://www.safaribooksonline.com>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F. et al. (Ed.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012. p. 1097–1105. Disponível em: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.

KUMAR, M.; DESHPANDE, A. *Artificial Intelligence for Big Data*. Packt, 2018. ISBN 9781788472173. Disponível em: <https://www.safaribooksonline.com>.

LI, F. F.; J., J.; S., Y. *CS231n: Convolutional Neural Networks for Visual Recognition*. 2017.

LI, S. Z. et al. Feature learning based on sae-pca network for human gesture recognition in rgbd images. *Neurocomputing*, v. 151, p. 565–573, 2015. ISSN 0925-2312. Disponível em: <<http>://WOS:000347753500005>.

MATLAB. *Matlab*. 2017. Disponível em: <https://www.mathworks.com/>.

MENSHAWY, A. *Deep Learning By Example*. Packt, 2018. ISBN 9781788399906. Disponível em: <https://www.safaribooksonline.com>.

PATTANAYAK, S. *Pro Deep Learning with TensorFlow: A Mathematical Approach to Advanced Artificial Intelligence in Python*. Apress, 2017. ISBN 9781484230961. Disponível em: <https://www.safaribooksonline.com>.

PATTERSON, J.; GIBSON, A. *Deep Learning: A Practitioner's Approach*. O'Reilly Media, 2017. ISBN 9781491914236. Disponível em: <https://www.safaribooksonline.com>.

PATTERSON, J.; GIBSON, A. *Getting started with deep learning*. O'Reilly Media, 2018. ISBN 9781492037323. Disponível em: <https://www.safaribooksonline.com>.

PIGOU, L. et al. Sign language recognition using convolutional neural networks. In: ____. *Computer Vision - Eccv 2014 Workshops, Pt I*. [s.n.], 2015. (Lecture Notes in Computer Science, v. 8925), p. 572–578. ISBN 978-3-319-16178-5; 978-3-319-16177-8. Disponível em: <<http>://WOS:000362493800040>.

PORFIRIO, A. J. et al. Libras sign language hand configuration recognition based on 3d meshes. In: ____. *2013 Ieee International Conference on Systems, Man, and Cybernetics*. [s.n.], 2013. (IEEE International Conference on Systems Man and Cybernetics Conference Proceedings), p. 1588–1593. ISBN 978-1-4799-0652-9. Disponível em: <<http>://WOS:000332201901120>.

PUGEAULT, N.; BOWDEN, R. Spelling it out: Real-time ASL fingerspelling recognition. In: *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain, November 6-13, 2011*. [s.n.], 2011. p. 1114–1119. Disponível em: <https://doi.org/10.1109/ICCVW.2011.6130290>.

STIEHL, D. et al. Towards a signwriting recognition system. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. [S.l.: s.n.], 2015. p. 26–30.

SUGOMORI, Y. et al. *Deep Learning: Practical Neural Networks with Java*. Packt, 2017. ISBN 9781788470315. Disponível em: <https://www.safaribooksonline.com>.

TOK, W.; DEAN, D.; M., S. *Deep Learning with Azure: Building and Deploying Artificial Intelligence Solutions on the Microsoft AI Platform*. Apress, 2018. ISBN 9781484236796. Disponível em: <https://www.safaribooksonline.com>.

TORRES, J. *Deep Learning, Introducción práctica con Keras*. Lulu Press, 2018. ISBN 9780244078959. Disponível em: <https://torres.ai/artificial-intelligence-content/deeplearning/>.

ZHU, X. et al. A two-stage detector for hand detection in ego-centric videos. In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. [S.l.: s.n.], 2016. p. 1–8.

APÊNDICE A – PESQUISA BIBLIOGRÁFICA SISTEMÁTICA

Neste apêndice é apresentada a pesquisa bibliográfica sistemática realizada em duas fases: planejamento e condução. A fase de planejamento é apresentada nas Tabelas 13 e 14 e o resultado da fase de condução é apresentado na Figura 31. A Tabela 15 apresenta os critérios estabelecidos nas etapas de triagem e análise dos artigos realizadas da fase de condução.

Tabela 13 – Planejamento da revisão

AUTOR:
Anne de Souza Oliveira
TEMA:
Utilização de Redes Neurais Convolucionais para Reconhecimento Automatizado da Língua de Sinais
QUESTÃO(ÕES) DA PESQUISA:
(1) Existem trabalhos utilizando redes neurais convolucionais para reconhecer a língua brasileira de sinais?
(2) Que técnicas para reconhecer a língua brasileira de sinais com imagens de profundidade estão sendo usadas?
(3) Existem trabalhos usando redes neurais convolucionais para reconhecer outras línguas de sinais que não sejam a brasileira com imagens de profundidade?
BASE DE DADOS NAS QUAIS A PESQUISA SERÁ CONDUZIDA:
IEE
Web of Science
Engineering Village
PALAVRAS CHAVE E ESTRATÉGIA DE BUSCA:
((convolutional neural network) AND ((brazilian sign language) OR (Libras)))
((depth image) AND ((Brazilian Sign Language) OR (Libras)))
((convolutional neural network) AND (sign language) AND (depth image))

Tabela 14 – Tabela de adequação das estratégias de busca a cada uma das bases de dados

FONTE	FERRAMENTA DE PESQUISA	COMPOSIÇÃO DA PESQUISA	FILTROS ADICIONAIS
IEE	Avançada	(("convolutional neural network") AND (("Brazilian Sign Language") OR ("Libras"))) (("depth image") AND (("Brazilian Sign Language") OR ("Libras"))) (("convolutional neural network") AND ("sign language") AND ("depth image"))	1 - Abrir a aba *Command Search* 2 - Selecionar no campo *Search* a opção *Ful Text & Metadata*
Web of Science	Avançada	TS=(convolutional neural network AND sign language) AND AD=(Brazil)	-
	Básica	(depth image) AND (brazilian sign language OR Libras) convolutional neural network AND sign language	Tópico
Engineering Village	Especializada	"convolutional neural network"wn ALL AND (Brazil* sign language* wn ALL OR Libras wn ALL) "depth"wn ALL AND (Brazil sign language wn ALL OR Libras wn ALL) "convolutional neural network"wn ALL AND "sign language"wn ALL AND "depth"wn ALL	-

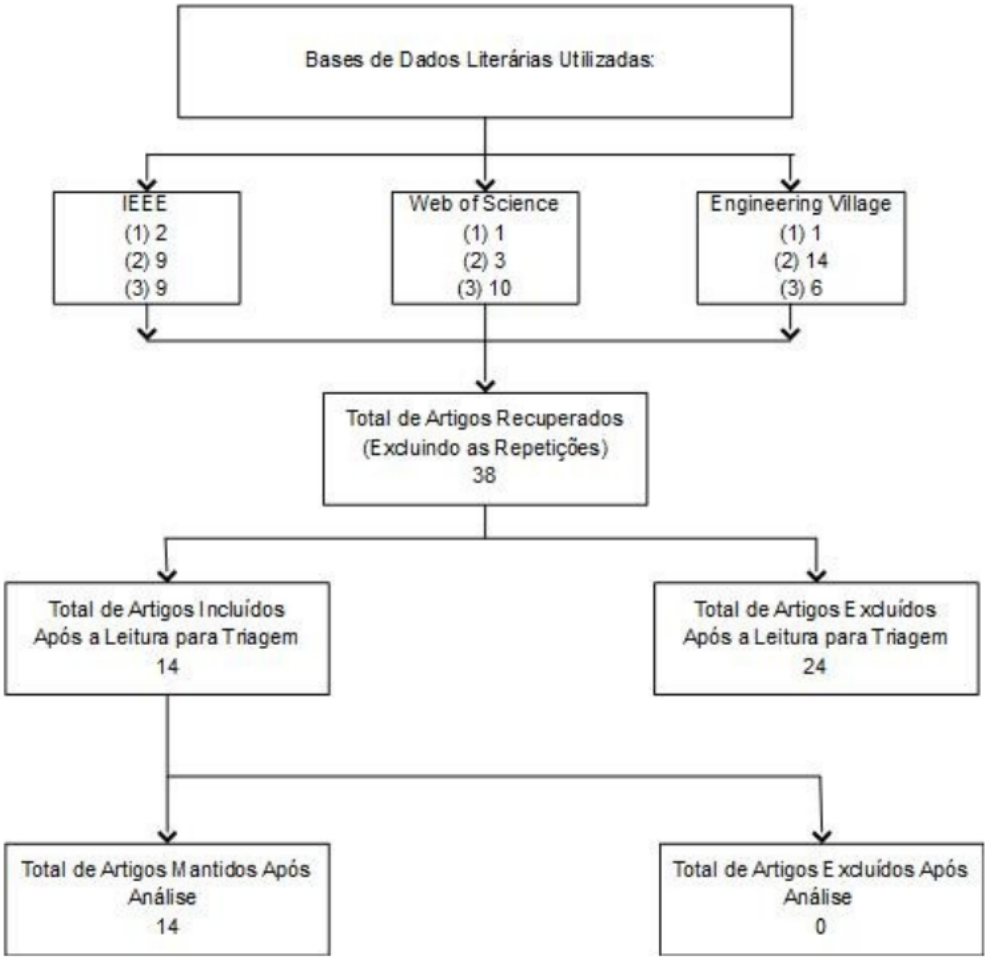


Figura 31 – Fluxograma da fase de seleção dos artigos.

Tabela 15 – Critérios utilizados na triagem e análise dos artigos

CRITÉRIOS DE INCLUSÃO/EXCLUSÃO UTILIZADOS NA TRIAGEM
(1) Os artigos precisavam citar a utilização de redes neurais convolucionais para reconhecimento da língua brasileira de sinais.
(2) Os artigos precisavam citar a utilização de amostras em profundidade no reconhecimento da língua brasileira de sinais.
(3) Os artigos precisavam citar a utilização de redes neurais convolucionais para reconhecimento de outras línguas de sinais com amostras em profundidade.
CRITÉRIOS DE INCLUSÃO/EXCLUSÃO UTILIZADOS NA FASE DE ANÁLISE DOS ARTIGOS (SE EXISTIREM)
Não houve critérios de inclusão/exclusão utilizados na fase de análise porque todos os artigos respondiam a pelo menos um dos questionamentos realizados no início da pesquisa.

APÊNDICE B – MATRIZ DE CONFUSÃO

Neste apêndice são apresentadas as matrizes de confusão dos três modelos com maior acurácia para reconhecer as configurações de mão da língua brasileira de sinais, cada um originalmente treinado por uma das arquiteturas utilizadas neste trabalho. As matrizes dos modelos #4, #8 e #12 estão presentes nas Figuras 32, 33 e 34.

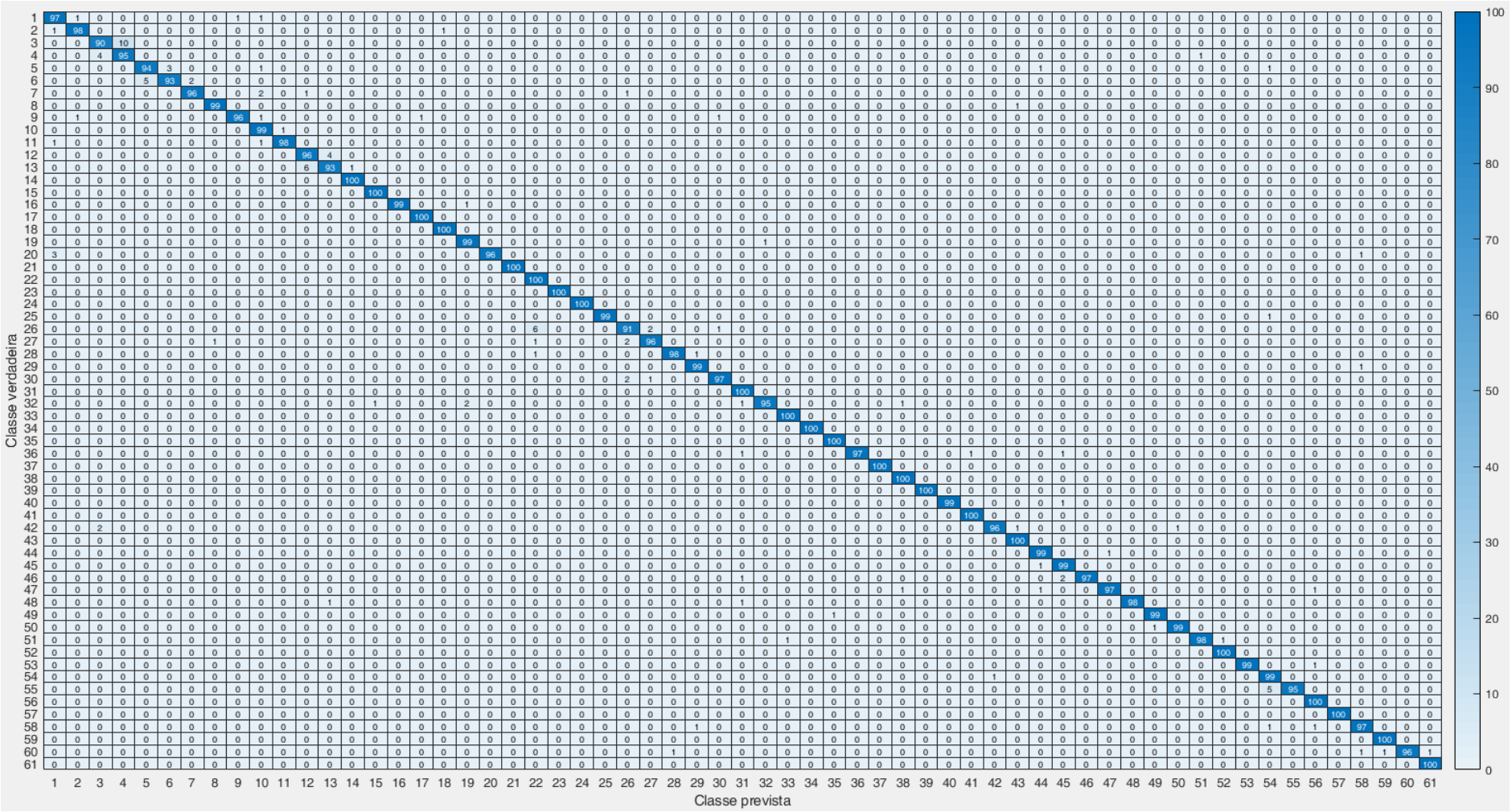


Figura 32 – Matriz de confusão do modelo #4.

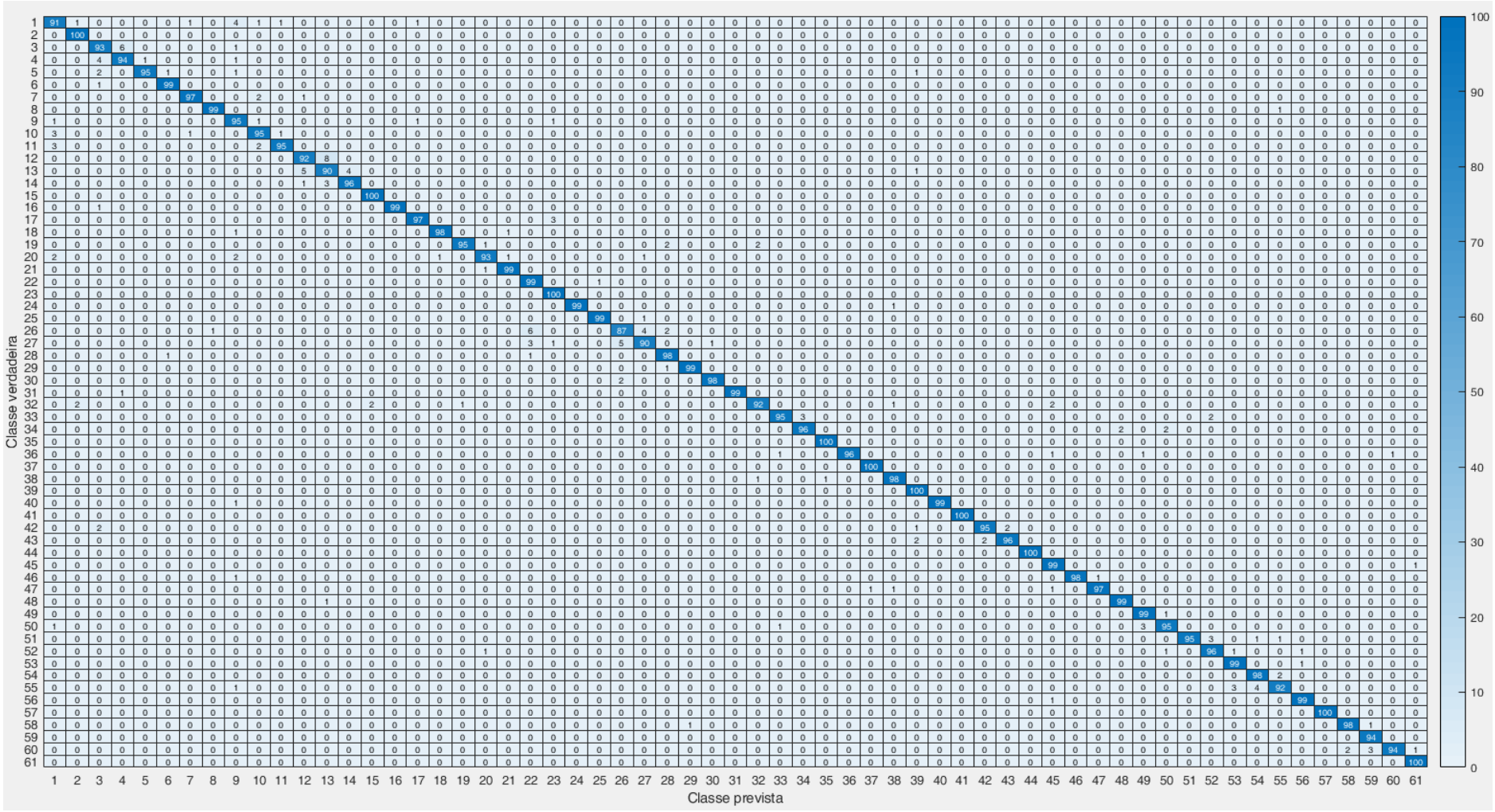


Figura 33 – Matriz de confusão do modelo #8.

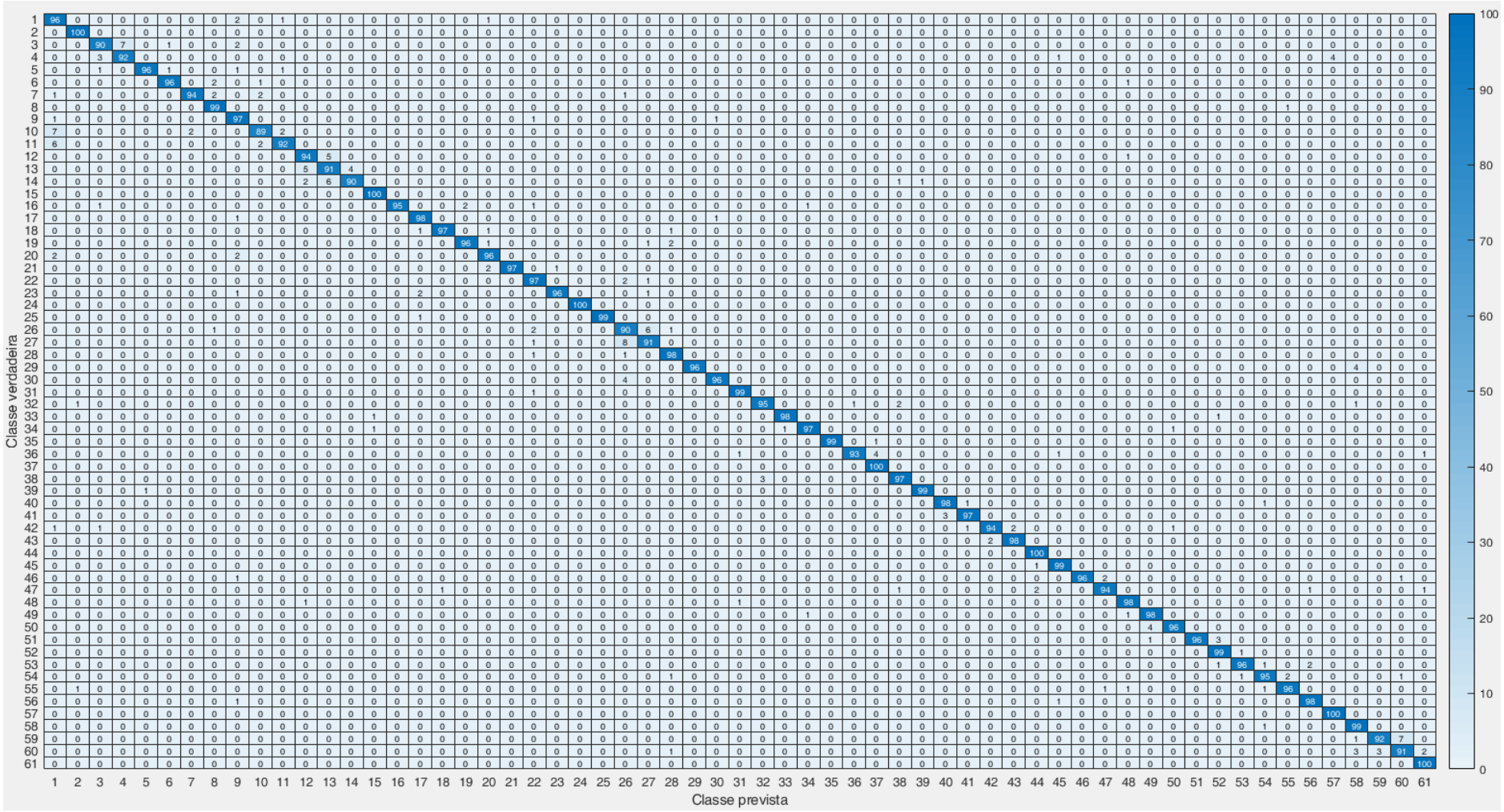


Figura 34 – Matriz de confusão do modelo #12.

APÊNDICE C – ARTIGO

Neste apêndice é apresentada a cópia do artigo originado deste trabalho. O artigo intitulado: *"Recognizing Hand Configurations of Brazilian Sign Language using Convolutional Neural Networks"*, foi apresentado no XXVI Congresso Brasileiro de Engenharia Biomédica (CBEB 2018), realizado na cidade de Búzios, Rio de Janeiro, de 21 a 25 de outubro de 2018.

Recognizing Hand Configurations of Brazilian Sign Language using Convolutional Neural Networks

A. S. Oliveria[1], C. F. F. Costa Filho[2] and M. G. F. Costa[3]
[1,2,3]Centro de P&D de Tecnologia Eletrônica e da Informação – CETELI
Universidade Federal do Amazonas, Manaus, Brasil
cffcfilho@gmail.com

Abstract. This paper proposes evaluating three convolutional neural network architectures for recognizing hand configurations of the Brazilian Sign Language (Libras). To improve the generalization of neural networks, two techniques were employed: dropout and L^2 regularization. A proprietary database consisting of 12.200 depth images, captured with the Kinect® sensor was used. Two hundred images were captured for each one of 61 Hand Configurations (HC) of Libras. The training and testing subsets were compounded using an interleave technique. An accuracy of 98% was achieved. This value is better than previous results obtained, with the same dataset, using the k-Nearest Neighbor (kNN) and Novelty classifiers, 95.41% and 96.31%, respectively.

Keywords: Deaf community, Sign language, Hand configuration, Gesture recognition, Convolutional Neural Network.

1 Introduction

Different models of machine learning have been employed to recognize the Brazilian Sign Language (Libras). Costa Filho *et al*. [1] and [2] applied the novelty classifier and the k-nearest neighbor classifier in a proprietary database of images captured with a Kinect® sensor. The image dataset constructed, called *LibrasImages* comprises 12,200 depth images. In depth images, the gray level of a pixel is proportional to the distance of an object to the camera. Two hundred images were captured for each one of 61 Hand Configurations (HC) of Libras from 10 volunteers. The techniques (2D)2LDA and (2D)2PCA were employed for feature extraction. In the former paper and in the last one, the authors obtained an accuracy of 95.41% and 96.31%, respectively.

Escobedo and Camara [3] and [4] applied a support vector machine with linear kernel and with an RBF kernel, respectively, in a proprietary database of images captures with a Kinect® sensor. The image database constructed has both depth and skeleton images. The database contains images of 18 HC of Libras. Two individuals repeated each gesture 20 times. In both papers, the authors extracted local and global characteristics. The database was randomly divided into 60% for training and 40% for testing. In the former paper and in the last one, an accuracy of 99.84% and 98.28%, respectively, was obtained. In both papers, the best results were obtained with a combination of local and global features.

Porfirio *et al.* [5] applied support vector machine with linear kernel to recognize 61 HC of Libras captured with a Kinect sensor. The image database constructed has both

depth and skeleton images. Five individuals repeated each gesture two times. In this study, the classification features were obtained from 3D mesh of the hands associated with features obtained from 2D images, corresponding to a frontal and a lateral view of the hand. The 2D extracted features were the following: seven Hu invariant moments, eight Freeman directions, and horizontal and vertical histogram projections. The features obtained from 3D mesh are some 3D mesh descriptors. The authors claim that these features are scale, rotation and translation invariant. The dataset was randomly separated in 70% for training and 30% for testing. An accuracy of 86.06% was obtained.

Almeida *et.al.* [6] applied support vector machine with RBF kernel to recognize 61 HC of Libras captured with a Kinect sensor. The image database constructed has both depth and skeleton images. One individual repeated each gesture five times. The authors presented a methodology for feature extraction in Libras that explores the phonological structure of the language and relies on RGB-D sensor for obtaining intensity, position and depth data. From the RGB-D images, seven vision-based features were obtained. Each feature is related to one, two or three structural elements in Libras. The dataset was randomly separated in 60% for training and 40% for testing. The accuracy rate was calculated for each gesture. A mean accuracy value higher than 80% was obtained.

Stiehl *et al.* [7] are the authors of the only paper that uses Convolutional Neural Network (CNN) to recognize HC of Libras. The authors employed 103 symbols of SignWriting system [8] to represent the 61 HC of Libras. The dataset contains 7997 gestures that were obtained from deaf and non-deaf people. To improve the generalization the authors used the technique of data augmentation. The images were rotated, smoothed, through the application of a mean filter and opening and closing morphological operations. The best accuracy reported by the authors was 94.4%.

Other authors applied CNN to recognize sign languages of other countries: Kang et al. [9] and Zhu [10] used CNN to recognize gestures of American Sign Language. Pigou *et al*. [11] used CNN to recognize gestures of Italian Sign Language.

In this paper we propose evaluating the performance of three architectures for recognizing the 61 HC of Libras. Architecture #1 is the *Alexnet* architecture [12]. This architecture was successfully proposed to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. Architectures #2 is adapted from architectures proposed in [10]. The difference is in the first layer, whose parameters have been adapted to work with images of different sizes. Architectures #3 is adapted from architectures proposed in [11]. The difference is that the architecture used is this paper is only a subset of the architecture used in [11]. The architecture used in [11] has two parallel branches, one for extracting hand characteristics and the other two extract body characteristics. We used only the former branch.

To improve the network performances, we also evaluate two generalization techniques, the dropout and the L^2 regularization.

2 Materials

The image database used in this paper is the same one used in [1] and [2]. The image database constructed, called *LibrasImages*, comprises 12,200 images. Two hundred images were captured for each one of 61 HC of Libras. Figure 1 shows these configurations. The images were captured from 10 volunteers, using a depth-sensing camera, the Kinect®. The image dimension is 640x480 elements and depth resolution is 11 bits. To obtain the 200 images of each gesture, captured from video frames, the volunteers are free to do the gesture in different positions relative to the body (articulation point), and to rotate the gesture from 45º to 135º, as illustrated in Figure 2. The scene illumination is not controlled.

Fig.1. Image showing the 61 HC of Libras

Fig.2. Examples of HC images acquired in different hand orientations.

The original dataset was divided in training and testing subsets, according to the following percentage: 50% for training and 50% for testing. In order to obtain data from all individuals in training and testing data subset, we employed the interleave technique. As shown in Figure 3, in this technique, a sample is selected to the training subset and the next one is selected to the testing subset. This procedure is repeated until there are no more samples.

4

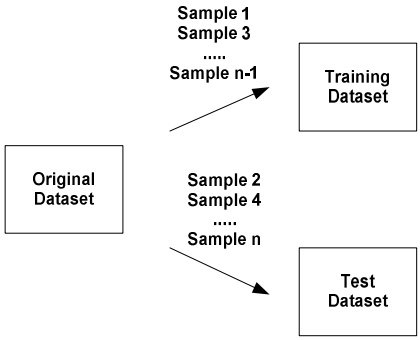


Fig. 3. Interleave technique to compound the training and testing subsets

3 Methods

Figure 4(a) shows architecture #1, the *Alexnet* architecture [12]. This is the more complex one and is comprised of 5 convolutional layers and 3 full connected layers. After each convolutional layer, there is a Rectifier Linear Unit layer (RELU layer). After the RELU layers 1 and 2, there are a maxpool layer and a cross normalization layer. After the RELU layer 5, there is a maxpool layer. After the full-connected layers 1 and 2, there are a RELU layer and a dropout layer. In the network output, there are a softmax and a classification layer.

Figure 4(b) shows architecture #2. This architecture, the simplest one, comprises 2 convolutional layers and 2 full connected layers. After each convolutional layer, there is a RELU layer and a maxpool layer. After the first full-connected layer, there is a RELU layer. In the network output, there are a softmax and a classification layer.

Figure 4(c) shows architecture #3. This architecture, with intermediary complexity, comprises 3 convolutional layers and 2 full connected layers. After the two first convolutional layers, there are a RELU layer, a cross normalization layer and a maxpool layer. In the sequence, there are a third convolutional layer, a RELU layer and a maxpool layer. Finally, there are a full-connected layer, a RELU layer and a dropout layer. In the network output, there are a softmax and a classification layer.

To improve neural network generalization, two techniques are employed: dropout and L^2 regularization. The dropout technique reduces overfitting by preventing complex co-adaptations on training data. A dropout ratio parameter sets the probability that any given unit is dropped. In this work, the dropout ratio parameter was set to 0.5, which is a value that has enabled the achievement of good results.

With the L^2 regularization criterion, the network error E_R is evaluated according to Equation (1).

$$E_R = \gamma E_{QM} + (1-\gamma)\sum w^2 \tag{1}$$

where:

E_{QM} –mean square error, Equation (2)

$$\frac{\sum_{n=1}^{N}(O_n - E_n)^2}{N} \tag{2}$$

O_n - Value obtained in the network output when sample *n* is presented in the input;
E_n – Expected value in the network output when sample *n* is presented in the input;
$\sum w^2$ – sum-of-square of the neural network weights;
γ – , L^2 regularization factor, 0<γ<1. In this study γ=0.2;

Concerning generalization, the experiments with the three architecture are the following: 1) simulation only with dropout layer and 2) simulation with dropout layer and L^2 regularization. In the *Alexnet* architecture, two dropout layers are already present, after the two full-connected layers. In architecture #2, a dropout layer is inserted after the second full-connected layer. In architecture #3, a dropout layer is already present after the first full-connected layer.

The networks were trained with the stochastic gradient descent with momentum (SGDM) algorithm. Three learning rates, *lr*, were employed: 0.1; 0.01 and 0.001. The training stop criterion was fixed in 500 epochs.

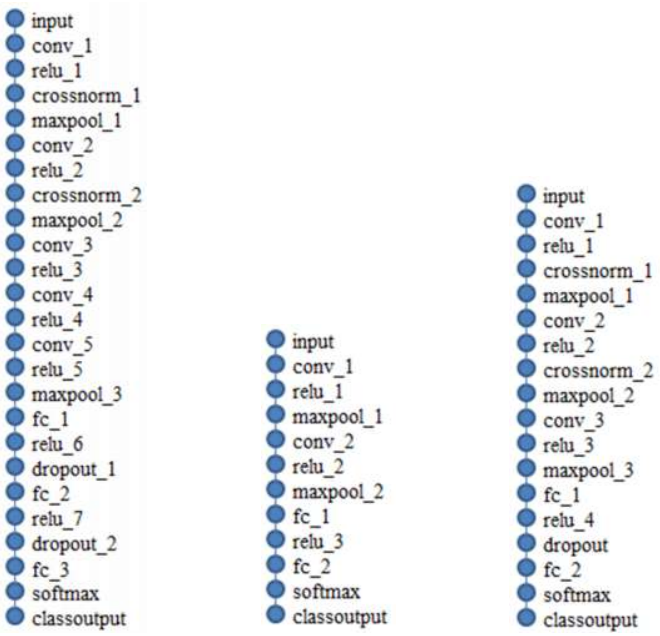


Fig. 4. (a) Architecture #1: *Alexnet* architecture [12]; (b) architecture #2: adapted from architectures proposed in [10]; (c) architecture #3: adapted from architecture proposed in [11].

6

3.1 Implementation

The simulations were made using a computer with Intel(R) Core (TM) i7, 3.60GHz Processor, 16GB of RAM, GPU NVIDIA, model GTX 745 with 4GB of RAM and 384 cuda cores, running Matlab 2017b.

4 Results

Figure 5 shows the learning curve for the three values of learning rate: 0.1, 0.01 and 0.001. As observed, with architecture #1, only the training with learning rates equal to 0.01 and 0.001 converged. With architecture #2, only the training with learning rate equal to 0.001 converged. Finally, with architecture #3, training with all learning rates converged. Aiming a fast training, we used a learning rate of 0.01 for architectures #1 and #3 and a learning rate of 0.001 for architecture #2.

In Table 1, we show the accuracy values obtained for all three architectures and techniques for improving generalization. As observed, for all architectures, the best values are obtained using dropout and L^2 regularization techniques together. The best accuracy, 97.98%, is obtained with architecture #1.

For architecture #1, over 22 hand configurations, the network accuracy was 100%. There were few problems when recognizing the other HC. The critical hand configurations were the 3, 6, 13 and 26. Figure 6(a) shows the more frequent problems when recognizing these configurations. Configuration 3 was misclassified as configuration 4 for 10% of the samples. Configuration 6 was misclassified as configuration 5 for 5% of the samples. Configuration 13 was misclassified as configuration 12 for 6% of the samples. Finally, configuration 26 was misclassified as configuration 22 for 5% of the samples.

For architecture #2, over 10 HC, the network accuracy was 100%. There were few problems when recognizing the other HC. The critical hand configurations were the 1, 13, 26 and 27. Figure 6(b) shows the more frequent problems when recognizing these configurations. Configuration 1 was misclassified as configuration 9 for 4% of the samples. Configuration 13 was misclassified as configuration 12 for 5% of the samples. Configuration 26 was misclassified as configuration 22 for 6% of the samples. Finally, configuration 26 was misclassified as configuration 22 for 5% of the samples.

For architecture #3, over 7 configurations, the network accuracy was 100%. There were few problems when recognizing the other configurations. The critical configurations were the 3, 10, 14 and 26. Figure 6(c) shows the more frequent problems when recognizing these configurations. Configuration 3 was misclassified as configuration 4 for 7% of the samples. Configuration 10 was misclassified as configuration 1 for 7% of the samples. Configuration 14 was misclassified as configuration 13 for 6% of the samples. Finally, configuration 26 was misclassified as configuration 27 for 6% of the samples.

Table 1. Accuracy values for the three architectures and techniques used for generalization improvement.

Architecture	Techniques for improving generalization	Accuracy (%)
Architecture #1	dropout	97.97
	dropout+ L^2 regularization	97.98
Architecture #2	dropout	96.52
	dropout + L^2 regularization	96.70
Architecture #3	dropout	85.41
	dropout+ L^2 regularization	96.15

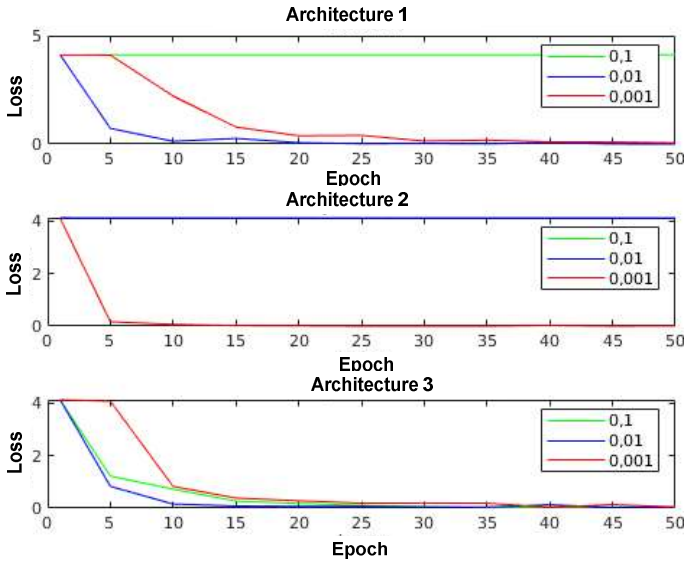


Fig. 5. Learning curve for the three architectures *versus* three values of learning rate: 0.1; 0.01 and 0.001.

8

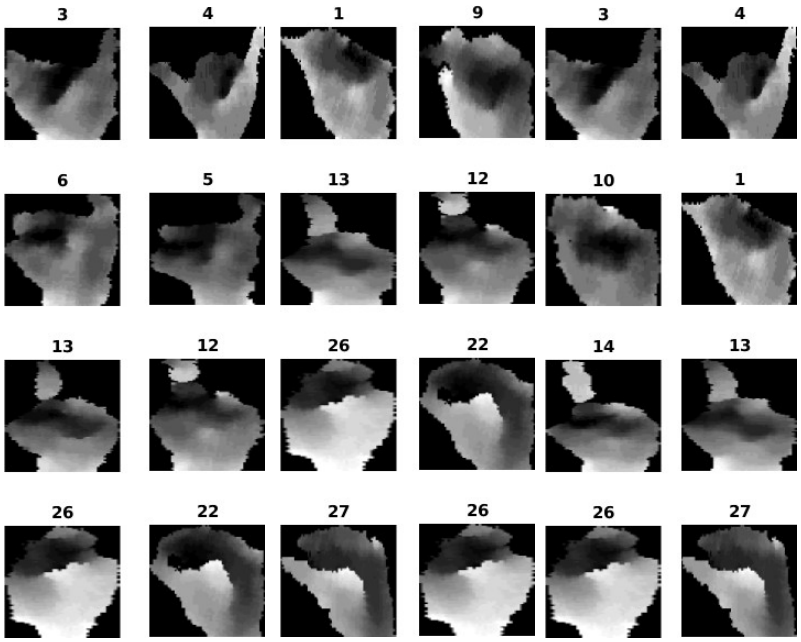


Fig. 6. More frequent errors when recognizing the 61 HC. (a) the first column shows a true configuration, while the second column shows the configuration recognized by architecture #1; (b) the first column shows a true configuration, while the second column shows the configuration recognized by architecture #2; (a) the first column shows a true configuration, while the second column shows the configuration recognized by architecture #3.

5 Conclusion

Differently from other studies previously published for recognizing Libras gestures, this one does not extract characteristics from each gesture as inputs to the neural network.

Methods that use depth maps, as the one proposed in this study, make a major contribution to this research field, since they do not depend on controlled environments [13,14,15] and do not require wearable sensors such as gloves with sensors [16,17].

In previous studies [1-2], using the same dataset, and applying the novelty classifier and the k-nearest neighbor classifier, we obtained an accuracy of 95.41% and 96.31%, respectively. In the present study, using convolutional neural networks, we obtained an accuracy of 97.98%. This result was obtained with an *Alexnet* architecture. With all three architectures studied, better results are obtained when we associate two techniques for generalization improvement, dropout and L^2 regularization.

The best gesture recognition accuracy of Libras obtained in this study, 97.98%, is much higher than the one obtained in Porfírio *et al*. [5], 86.06%. It must be emphasized

that this recognition rate is obtained for different conditions of hand rotation and proximity of the depth camera, and with a depth camera resolution of only 640x480 pixels.

Although a high accuracy was obtained, some recognizing errors were verified in specific HC's. These errors are due to the similarity between some HC's, as shown in Figure 6. In future, to minimize theses errors, we intend to associate other techniques to improve generalization, as data augmentation and early stopping.

Acknowledgments

The authors would like to thank FAPEAM – PROTI Amazônia Pesquisa (Edital 011/2017) and CNPq – INCT-MACC -2015/2021(Instituto Nacional de Ciência e Tecnologia – Medicina Assistida por Computação Científica.

References

1. Costa Filho, C.F.F., Santos, B.L., Souza, R.S., Santos, J.R., Costa, M.G.F.: A new method for recognizing hand configurations of brazilian gesture language. In: 38th Annual International Conference of the IEEE-Engineering-in-Medicine-and- Biology-Society (EMBC), IEEE Engineering in Medicine and Biology Society Conference Proceedings, pp. 3829-3834 (2016).
2. Costa Filho, C.F.F, Santos, B.L., Souza, R.S., Santos, J.R., Costa, M.G.F.: A fully automatic method for recognizing hand configurations of Brazilian Sign Language. Research on Biomedical Engineering, 33(1), 78-89 (2017).
3. Esobedo, E.C. and Camara, G.C.: A new approach for dynamic gesture recognition using skeleton trajectory representation and histograms of cumulative magnitudes. In 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 209-216 (2016).
4. Escobedo, E.C., Camara, G.C.: A robust gesture recognition using hand local data and skeleton trajectory. In: IEEE International Conference on Image Processing (ICIP), pp. 1240-1244 (2015).
5. Porfirio A.J., Wiggers K.L., Oliveira, L.E.S. and Weingaertner D.: Libras Sign Language Hand Configuration Recognition Based on 3D Meshes. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, Manchester, 1588-1593 (2013).
6. Almeida, S.G.M., Guimaraes, F.G., Ramirez, J.A.: Feature extraction in brazilian sign language recognition based on phonological structure and using rgb-d sensors. Expert Systems with Applications, 41(16), 7259-7271 (2014).
7. Stiehl, D., Addams, L., Oliveira, L.S., Guimaraes, C. and Brito Jr, A.S.: Towards a signwriting recognition system. In: 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 26-30 (2015).
8. Sutton, Y.: A way to analyze american sign language and any other sign language without translation into any spoken language. In National Symposium on Sign Language Research and Teaching (1980).
9. Kang, B., Tripathi, S. and Nguyen, T.Q.: Real-time sign language fingerspelling recognition using convolutional neural networks from depth map. In: 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pp. 136-140 (2015).

10

10. Zhu, X., Liu, W., Jia, X. and Wong, K.Y. K.: A two-stage detector for hand detection in ego-centric videos. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1-8 (2016).
11. Pigou, L., Dieleman, S., Kindermans, P.J. and Schrauwen, B.: Sign language recognition using convolutional neural networks. In: Computer Vision - Eccv 2014 Workshops, (Lecture Notes in Computer Science, vol. 8925, pp. 572-578 (2015).
12. Krizhevsky, A., Sutskever, I. and Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: PEREIRA, F. et al. (Ed.). Advances in Neural Information Processing Systems 25. Curran Associates, Inc., pp. 1097-1105 (2012).
13. Neris, M.N., Silva, A.J., Peres, S.M., Flores, F.C.: Self organizing maps and bit signature: a study applied on signal language recognition. In: IEEE International Joint Conference on Neural Networks. 2008. pp. 2934-941 (2008).
14. Carneiro, T.S., Cortez, P.C., Costa, R.C.S.: Recognizing Libras gestures with neural classifiers, using Hu invariant moments. In: Interaction 09 - South America, São Paulo, pp. 190-195 (2009).
15. Pizzolato, E.B., Anjo, M.S., Pedroso, G.C.: Automatic Recognition of Finger Spelling for Libras based on a Two Layer Architecture. In: ACM Symposium on Applied Computing. Proceedings of the 25th Symposium on Applied Computing. pp. 970-74 (2010).
16. Mehdi, S.A., Khan, Y.N.: Sign language recognition using sensor gloves. In IEEE Neural Information Processing, ICONIP'02. Proceedings of the 9th International Conference. vol. 5, pp. 2204-2206 (2002).
17. Wang, H.: Nearest neighbors by neighborhood counting, IEEE Transactions on Pattern Analysis and Machine Intelligence. 28(6), 942-953 (2006).